UNIVERSITADE DO VALE DO TAQUARI
CURSO DE ENGENHARIA DE SOFTWARE

PROTOTIPAÇÃO DE UMA SOLUÇÃO PARA CONTROLE DE DISPOSITIVOS EM CASAS INTELIGENTES POR MEIO DO RECONHECIMENTO DE GESTOS ESTÁTICOS E BIOMETRIA FACIAL

Klaus Fernando Etgeton

Lajeado, Novembro de 2019

Klaus Fernando Etgeton

PROTOTIPAÇÃO DE UMA SOLUÇÃO PARA CONTROLE DE DISPOSITIVOS EM CASAS INTELIGENTES POR MEIO DO RECONHECIMENTO DE GESTOS ESTÁTICOS E BIOMETRIA FACIAL

Trabalho de Conclusão de Curso apresentado ao Centro de Ciências Exatas e Tecnológicas do Centro Universidade do Vale do Taquari – Univates, como parte dos requisitos para a obtenção do título de bacharel em Engenharia de Software.

Orientador: Prof. Msc Willian Valmorbida

Lajeado, Novembro de 2019

AGRADECIMENTOS

A minha família, amigos e minha namorada Caroline, pelo constante incentivo, apoio e compreensão pelos momentos de ausência, em virtude da dedicação necessária para a realização deste trabalho.

Ao professor Willian Valmorbida pelo apoio e orientação durante o desenvolvimento de todo o trabalho.

E a todos os demais que, de alguma maneira, contribuíram para a realização deste trabalho.

RESUMO

Estima-se que bilhões de dispositivos IoT estarão conectados nos próximos anos, criando inúmeras possibilidades para automação em casas inteligentes que poderá tornar a vida das pessoas mais prática, eficiente, confortável. Além das funcionalidades, as pessoas também esperam que as tecnologias se integrem e façam parte do ambiente, proporcionando melhores experiências de interação de maneira mais fácil e natural. Neste trabalho, foi desenvolvido um sistema de visão computacional em Python, com uso de modelos em Deep Learning, para controlar dispositivos IoT conectados em casas inteligentes por meio do reconhecimento de gestos estáticos da mão, realizando uma interação mais segura por meio da autenticação biométrica facial dos usuários. Para o reconhecimento em tempo real, o sistema apresenta um frame rate de 3fps. O reconhecimento facial dos usuários apresentou excelentes resultados, com apenas 1.10% de falso positivo, enquanto o reconhecimento de gestos estáticos apresentou 22.67% de falso positivo. O sistema proposto atende os requisitos necessários com desempenho relativamente bom, porém melhorias quanto ao reconhecimento de gestos estáticos ainda precisam ser realizadas para reduzir sua taxa de falso positivo.

Palavras-chave: Visão computacional. Casas inteligentes. Biometria facial. Reconhecimento de gestos. Deep Learning.

ABSTRACT

Billions of IoT devices are expected to be connected in the coming years, creating countless possibilities for smart home automation that could make people's lives more practical, efficient, comfortable. In addition to functionality, people also expect technologies to integrate and be part of the environment, providing better interaction experiences, more easily and naturally. In this work, a Python computational vision system was developed, using Deep Learning models, to control smart home connected IoT devices through the recognition of static hand gestures, performing a more secure interaction through facial biometric authentication of users. For real time recognition, the system has a frame rate of 3fps. Facial recognition of users showed excellent results, with only 1.10% false positive, while recognition of static gestures presented 22.67% of false positive. The proposed system meets the necessary requirements with relatively good performance, but improvements in static gesture recognition have yet to be made to reduce its false positive rate.

Keywords: Computer vision. Smart houses. Facial biometrics. Gesture recognition. Deep learning.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

API	Application Programming Interface
CPU	Central Processing Unit
GPU	Graphics Processing Unit
HOG	Histogram of Oriented Gradients
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
IOT	Internet of Things
JPEG	Joint Photographic Experts Group
LDA	Linear Discriminant Analysis
OCR	Optical Character Recognition
PIN	Personal Identification Number
RAM	Random Access Memory
REST	Representational State Transfer
RFID	Radio Frequency Identification
SQL	Structured Query Language
SVM	Support Vector Machine

SUMÁRIO

1 INTRODUÇÃO

Pode-se observar que nos últimos anos houve um investimento considerável em pesquisas e desenvolvimento de produtos que utilizam da visão computacional para a identificação, reconhecimento e classificação de informações em imagens. Há inúmeros projetos de pesquisa e ferramentas que propõem solucionar e criar modelos que consigam classificar e identificar desde pessoas, animais, placas de trânsito e até mesmo sentimentos.

A visão computacional possui inúmeras aplicações, tanto na indústria, ao realizar a inspeção de defeitos em placas eletrônicas na linha de produção, quanto no meio domiciliar, presente nas interfaces dos vídeos games, em câmeras e celulares ao remover os olhos vermelhos das pessoas presentes nas fotos (DAWSON-HOWE, 2014).

Tecnologias provenientes da visão computacional estão proporcionando diferentes meios para que possamos interagir e facilitar tarefas cotidianas, proporcionando redução de tempo e praticidade. A empresa Amazon, por exemplo, está trabalhando em um modelo de mercado inteligente, chamado de Amazon Go, no qual de maneira autônoma, pessoas possam realizar compras simplesmente ao adicionar os produtos no carrinho de compras, sem a necessidade de passar por filas, sendo que a cobrança ocorre depois de sair da loja mediante cartão de crédito pré-cadastrado. A identificação do que foi comprado ocorre com a utilização de diversos sensores que realizam o rastreamento e análise dos produtos que foram retirados das prateleiras. Este modo de compra dispensa pagamento por meio do contato físico, garantindo maior praticidade, menor número de interrupções e interação mais intuitiva e natural (REUTERS, 2018).

A criação de ambientes inteligentes e com maneiras práticas de interação se fazem possíveis graças ao aprimoramento de tecnologias portáteis e de baixo custo. Dispositivos dos

mais variados tipos agora possuem uma camada de comunicação que possibilita desde ações mais simples como ligar e desligar a até mais complexas como controlar a central de uma casa inteligente.

Nos últimos anos, grandes avanços ocorreram graças à criação de modelos e estruturas que conseguem extrair e identificar padrões de dados, possibilitando que máquinas consigam aprender. A aprendizagem pode ocorrer usando desde metodologias supervisionadas até não supervisionadas, para que dado um conjunto de dados, a máquina consiga criar um conjunto de regras (aprendizagem) e então passe a inferir respostas utilizando novos valores como entrada.

Até o ano 2021, estima-se que o número de dispositivos de Internet Of Things (IoT) conectados alcançará o patamar de 25 bilhões, estes dispositivos tornarão nossa vida mais prática, eficiente, econômica e confortável (GARTNER, 2018). Com o uso de assistentes pessoais treinadas com Machine Learning é possível que tarefas do cotidiano como reserva em restaurantes e controle das luzes em casa sejam automatizados.

Conforme Benyon (2014), as pessoas possuem certas expectativas quanto às casas inteligentes, elas geralmente não possuem muito tempo e buscam maneiras de otimizar a realização das tarefas do cotidiano, ao mesmo tempo que as casas inteligentes também precisam melhorar a experiência de quem vive nela. As tecnologias precisam se integrar e fazer parte do ambiente, com interfaces transparentes que não só focam em funcionalidades, mas também na experiência que elas podem agregar, criando uma interação mais fácil e natural.

A segurança é um ponto de interesse no contexto de interação com casas inteligentes, pois devemos impedir que pessoas não autorizadas assumam o controle de um dispositivo ou de uma casa, conforme Dickson (2016), muitos fabricantes ainda não possuem infraestrutura para soluções de segurança em dispositivos IoT.

Neste contexto, a biometria facial é uma das técnicas que pode ser utilizada para agregar autenticação, buscando certo nível de segurança na interação com casas inteligentes, esta técnica será utilizada no trabalho, meio de validar a permissão de uma pessoa ao realizar uma ação em um ambiente que deve ser controlado.

1.1 Objetivos Gerais

Este trabalho objetiva criar um protótipo de solução que possibilite o controle de dispositivos IoT conectados, em ambientes de casas inteligentes, por meio do reconhecimento de gestos estáticos da mão, garantindo a segurança por meio da autenticação da biometria facial dos usuários.

1.2 Objetivos Específicos

- Estudar e compreender a utilização e integração de tecnologias baseadas em visão computacional para a prototipação de uma solução de software;
- Especificar uma solução que integre tecnologias existentes em um sistema que possibilite realizar a autenticação de uma pessoa por meio da biometria facial, além do reconhecimento e execução de comandos por meio dos gestos estáticos das mãos com algum dispositivo IoT;
- Desenvolver um protótipo baseado na especificação da solução;
- Realizar testes no protótipo desenvolvido;
- Analisar os resultados obtidos com o protótipo, a fim de verificar a viabilidade da solução, analisando também as tecnologias utilizadas para resolução dos problemas.

1.3 Organização do trabalho

Os capítulos que compõe o presente trabalho estão organizados como segue:

- No Capítulo 2 é apresentado o referencial teórico, contendo a revisão dos conceitos que foram utilizados no trabalho, além dos trabalhos relacionados;

- No Capítulo 3 é apresentada a metodologia utilizada no trabalho, assim como o método no qual ele se baseia, neste capítulo também são apresentadas as tecnologias que foram utilizadas para o desenvolvimento do protótipo;

- No Capítulo 4 é apresentada a especificação do projeto, com o detalhamento dos pormenores do projeto;

- No Capítulo 5 é apresentado o desenvolvimento do protótipo com base na especificação;

- No Capítulo 6 são apresentados os resultados obtidos através de experimentos realizados com o protótipo.

- No Capítulo 7 é apresentada a conclusão e trabalhos futuros.

2 REFERENCIAL TEÓRICO

Para o desenvolvimento do protótipo da solução proposta, torna-se necessário o estudo de visão computacional para processamento de imagens, bem como modelos de identificação e extração da biometria facial, noção sobre Machine Learning, meios de interação com ambientes inteligentes e sua relação com dispositivos IoT, proporcionando uma interação mais natural e fluída. Estes assuntos serão abordados nas seções a seguir, visando exemplificar o contexto e os problemas que este trabalho se propõe a resolver.

2.1 Visão computacional

Conforme Dawson-Howe (2014, p. 19), visão computacional é a análise automática de imagens e vídeos a fim de obter uma compreensão do mundo. É baseada na capacidade de visão humana, sendo inicialmente abordada como tema na década de 60 e 70, pensada como um problema simples de ser resolvido, devido ao nosso próprio sistema visual que faz o processamento parecer algo intuitivo para a nossa mente. No entanto, o sistema visual humano é muito complexo e estima-se que nosso cérebro utiliza de 25% até mais de 50% de nossa capacidade para processar as imagens, realizando inúmeras interpretações ao visualizar uma imagem, como sentimentos, o vento sobre as folhas, interação entre pessoas.

O sistema visual humano utiliza inúmeras informações aprendidas durante a vida para interpretar e inferir alguma informação quando visualizada. A complexidade da visão computacional encontra-se em converter toda a noção intuitiva que um humano desenvolve ao longo da vida em um modelo que possa ser interpretado por um computador. Segundo Dawson-

Howe (2014 p. 19), para um computador uma imagem é apenas uma matriz de valores numéricos, como a representação de uma face de uma pessoa, que para nós humanos apenas conseguimos entender quando em formato de foto (Figura 1).

Figura 1 – Representação de uma foto em modelo computacional.

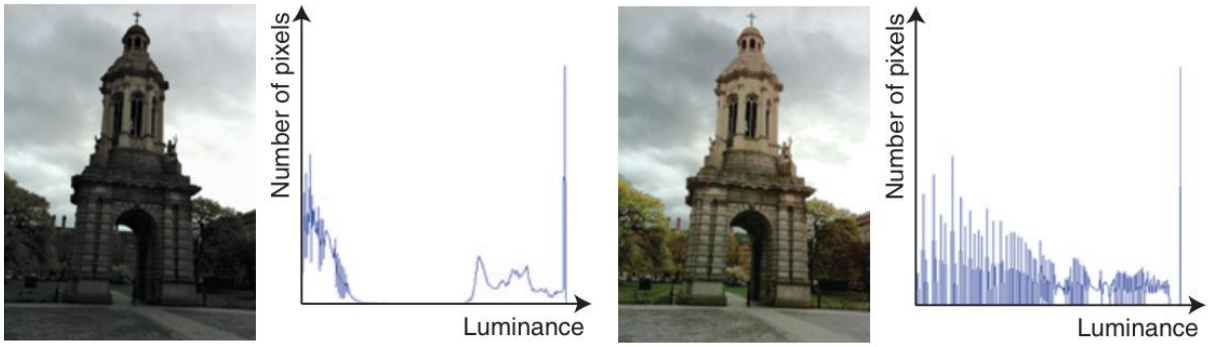
67	67	66	68	66	67	64	65	65	63	63	69	61	64	63	66	61	60
69	68	63	68	65	62	65	61	50	26	32	65	61	67	64	65	66	63
72	71	70	87	67	60	28	21	17	18	13	15	20	59	61	65	66	64
75	73	76	78	67	26	20	19	16	18	16	13	18	21	50	61	69	70
74	75	78	74	39	31	31	30	46	37	69	66	64	43	18	63	69	60
73	75	77	64	41	20	18	22	63	92	99	88	78	73	39	40	59	65
74	75	71	42	19	12	14	28	79	102	107	96	87	79	57	29	68	66
75	75	66	43	12	11	16	62	87	84	84	108	83	84	59	39	70	66
76	74	49	42	37	10	34	78	90	99	68	94	97	51	40	69	72	65
76	63	40	57	123	88	60	83	95	88	80	71	67	69	32	67	73	73
78	50	32	33	90	121	66	86	100	116	87	85	80	74	71	56	58	48
80	40	33	16	63	107	57	86	103	113	113	104	94	86	77	48	47	45
88	41	35	10	15	94	67	96	98	91	86	105	81	77	71	35	45	47
87	51	35	15	15	17	51	92	104	101	72	74	87	100	27	31	44	46
86	42	47	11	13	16	71	76	89	95	116	91	67	87	12	25	43	51
96	67	20	12	17	17	86	89	90	101	96	89	62	13	11	19	40	51
99	88	19	15	15	18	32	107	99	86	95	92	26	13	13	16	49	52
99	77	16	14	14	16	35	115	111	109	91	79	17	16	13	46	48	51

Fonte: Dawson-Howe (2014)

2.1.1 Equalização do histograma de cores de uma imagem

Há situações durante a visualização de imagens onde que há regiões muito claras e outras regiões da mesma muito escuras, há uma grande variedade de tonalidades desde as cores claras até as escuras que dificultam a compreensão de uma imagem, entretanto se for realizado a distribuição entre as escalas em cinza podemos compreender de maneira melhor o que está presente na imagem. Uma técnica que realiza esta operação de distribuição das escalas em cinza em uma imagem é a equalização do histograma, conforme a Figura 2, é possível verificar o resultado da equalização da escala em cinza no canal de luminosidade, e como resultado final, a imagem possuí maior contraste e níveis de luminosidade equilibrados (DAWSON-HOWE, 2014).

Figura 2 – Equalização do histograma de luminosidade de uma imagem



Fonte: Dawson-Howe (2014)

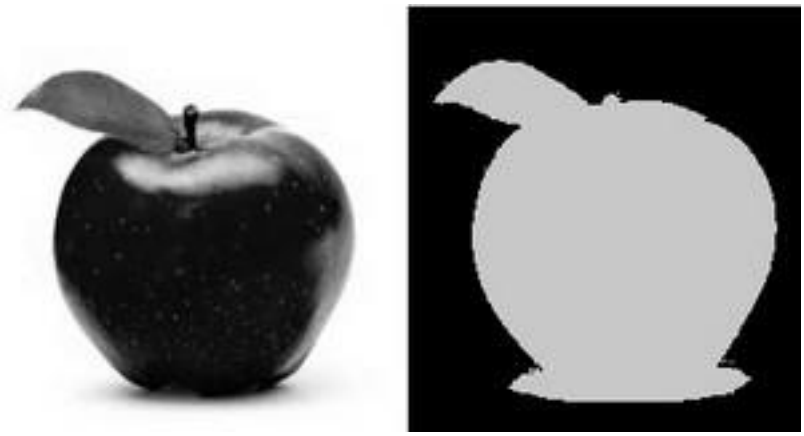
2.1.2 Segmentação

A segmentação de imagem é o processo de quebrar ou segmentar uma imagem em múltiplos grupos de *pixels* que geralmente possuem características em comum, tendo como principal objetivo representar a imagem de um modo que faça mais sentido e consequentemente facilitando sua análise. O processo de segmentação também pode ocorrer com base na similaridade de cor ou forma do conjunto de *pixels*. Nos últimos anos vários algoritmos e técnicas foram desenvolvidas com o uso de conhecimentos específicos para aumentar a eficiência na segmentação de imagens para problemas específicos, como na medicina, condução de veículos autônomos e vigilância (MATHWORKS, 2019).

O método mais simples de segmentação é o Thresholding que realiza a separação das regiões com base na variação de intensidade entre os *pixels* do objeto de interesse e os *pixels* do fundo da imagem. Para diferenciar os *pixels* do objeto de interesse é realizado a comparação da intensidade de cada *pixel* frente a um valor limite conhecido como *threshold* (OPENCV, 2019).

Para realizar a separação dos *pixels* da imagem são utilizados dois valores que podem variar de acordo com a necessidade, definindo um valor mínimo para os *pixels* que possuem valor menor que o *threshold* e assumindo o valor máximo para *pixels* de valor superior (OPENCV, 2019). Conforme a Figura 3, é possível verificar a segmentação de uma maçã, atribuindo-se 0 (preto) para os *pixels* de menor valor e 255 (branco) para os *pixels* que possuem valor superior ao *threshold* considerado.

Figura 3 – Segmentação de uma maçã

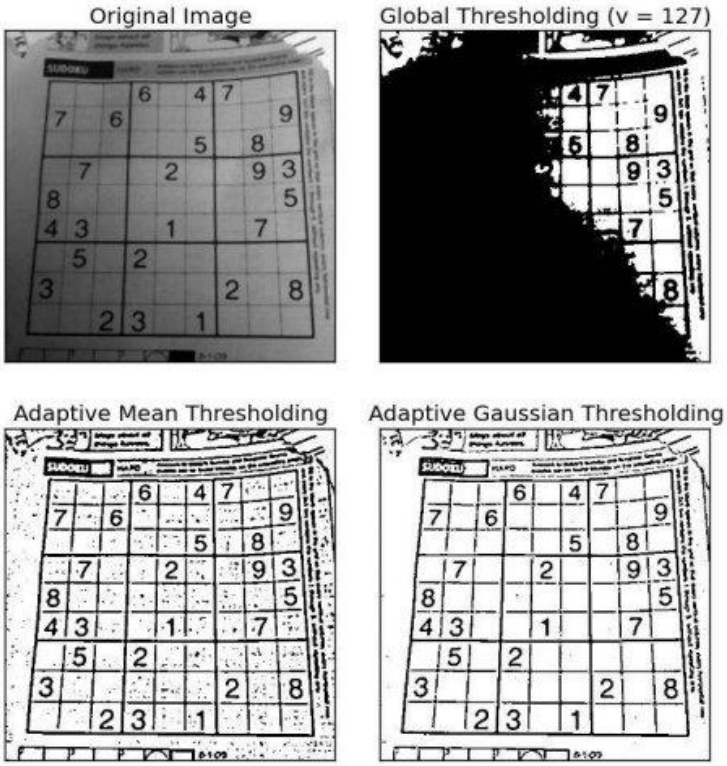


Fonte: OpenCV (2019)

O ponto negativo do Thresholding simples é a utilização de apenas um *threshold* global (Global Thresholding) para verificação de todos os *pixels* da imagem, não há distinção entre as diferentes regiões. Desta maneira, para imagens que apresentam diferentes condições de luminosidade o resultado final da segmentação pode não ser adequado (OPENCV, 2019).

Para resolver este problema pode ser utilizado um modelo que se adapte as diferentes condições de luminosidade que podem existir nas diferentes regiões de uma imagem, este modelo é conhecido como Adaptive Thresholding. Este algoritmo realiza a definição do *threshold* com base em uma pequena região que cerca cada *pixel*, assim obtendo melhores resultados para imagens com grande variação de luminosidade (OPENCV, 2019). Conforme Figura 4, é possível visualizar o resultado da aplicação de algoritmos baseados em Tresholding global e adaptativo, os melhores resultados para segmentação são claramente notáveis nos modelos adaptativos.

Figura 4 – Resultado da aplicação de diferentes algoritmos Thresholding



Fonte: OpenCV (2019)

2.1.3 Detecção de objetos com classificadores

O processo de detecção de objetos ou formas é comumente utilizado em inúmeras tarefas da computação para encontrar faces, pessoas, placas de trânsito. O algoritmo Haar proposto por Paul Viola e Michel Jones se demonstra como uma maneira efetiva para resolver o problema de detecção de objetos. O algoritmo é baseado em Machine Learning e para a criação de um modelo classificador é necessário o treinamento com uma grande quantidade de imagens positivas e negativas. Para a aprendizagem deste modelo são necessárias imagens positivas, ou seja, que representam o objeto de interesse de detecção e imagens negativas onde não há o objeto de interesse, desta maneira o modelo aprende a considerar quais são as características do objeto em questão (OPENCV, 2019).

Algoritmos de classificação baseados em Haar do OpenCV foram utilizados para a detecção da posição da palma e punho da mão.

2.2 Biometria

A biometria deriva do grego *bios* (vida) e *metron* (medida), em contexto de autenticação refere-se à capacidade em extrair características únicas de uma pessoa a fim de garantir uma entidade única e possibilitar sua autenticação (MAGALHÃES; SANTOS, 2003).

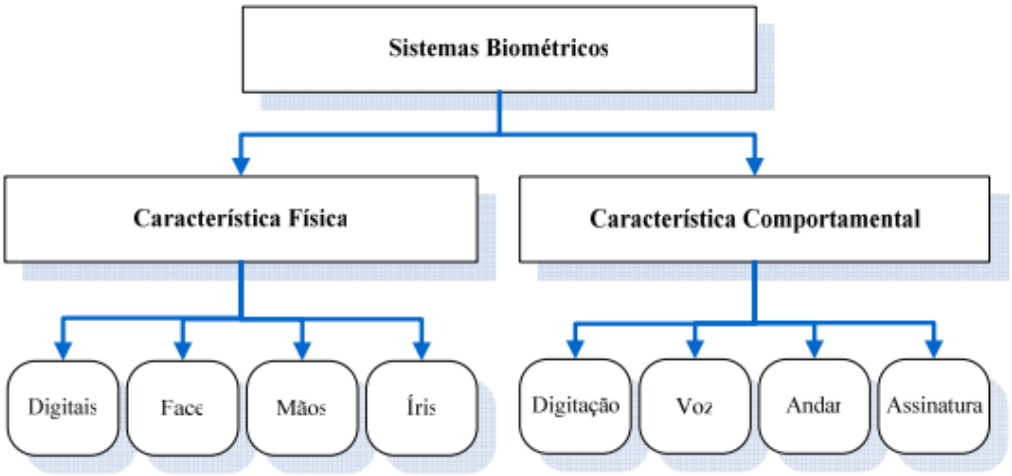
Segundo Nakashiro (2011), para se diferenciar uma pessoa de outra é necessário que exista um meio de definir sua identidade, ou seja, um conjunto de características únicas que se diferem de outra pessoa, como traços físicos, traços faciais, a íris, a retina, voz e grafia.

A biometria como meio de autenticação é utilizada há muito tempo, não é uma prática recente, segundo Watson (2008), os babilônios pressionavam a ponta dos dedos na argila como meio de registrar as transações comerciais.

Métodos convencionais de autenticação utilizam de uma informação conhecida como PIN, ou senha, entretanto informações deste tipo podem ser perdidas, esquecidas e até mesmo roubadas para que seja possível acessar um conteúdo restrito. Desta forma, cria-se uma motivação na utilização de técnicas de autenticação que utilizam da biometria (COSTA, 2001).

As técnicas de biometria são classificadas em dois tipos diferentes, as que consideram características físicas e as comportamentais, conforme apresentado na Figura 5. As características físicas são medidas estáveis e praticamente imutáveis ao longo da vida de uma pessoa, como face, impressão digital, íris, retina. As características comportamentais como fonte de medida são baseadas nas ações únicas de uma pessoa que foram adquiridas ao longo de sua vida, elas podem sofrer variações no padrão ao longo do tempo, como modo de andar, assinatura, voz (BOECHAT, 2008).

Figura 5 – Características biométricas



Fonte: Boechat (2008)

Tolba, El-Baz e El-Harby (2008) definem que de maneira geral, análises biométricas podem ser realizadas em três etapas, sendo elas:

1. Sensor biométrico captura uma assinatura biométrica da pessoa;

2. Um algoritmo computacional normaliza e limpa os dados que possam causar ruídos na identificação da assinatura biométrica, e possíveis conversões para adaptar o padrão biométrico, como tamanho, resolução, iluminação. Cria-se uma assinatura biométrica normalizada da pessoa;

3. E por último, utiliza-se um algoritmo de comparação, que dado o padrão normalizado da etapa 2, compara-se em uma base de dados com outros padrões também normalizados para criar uma taxa de similaridade.

Para que uma característica humana seja usada como unidade de identificação biométrica, é necessário que ela satisfaça um conjunto básico de requisitos, assim como referenciado por Boechat (2008). São eles:

- Universalidade: deve ser aplicável em todas as pessoas, ou seja, característica que possa ser encontrada em todos os seres humanos;

- Permanência ou imutabilidade: a característica deve se manter ao longo da vida, sem se modificar;

- Mensurabilidade: a característica precisa ser passível de medição quantitativa.

Para a utilização em modelos computacionais, as características biométricas, também precisam ser passíveis de serem utilizadas de maneira prática e rápida em algoritmos computacionais, como o desempenho em ser identificada e processada, tamanho necessário para armazenar o padrão biométrico em uma base de dados (BOECHAT, 2008).

Os sistemas de identificação que utilizam da característica biométrica como fator de autenticação, dificilmente conseguem 100% de certeza, pois algumas de nossas características biométricas podem variar e sofrer pequenas alterações ao longo de nossa vida, dificultando a comparação do padrão biométrico extraído. Como agravantes ao realizar a identificação de uma pessoa, Moraes (2010) menciona, na identificação facial a presença ou ausência de barba, óculos, acessórios.

Frente à variação que existe, é necessário determinar um grau de certeza para determinar se uma pessoa será ou não autenticada como verdadeira para obter acesso em um determinado contexto.

Sistemas biométricos de autenticação, frente a uma característica biométrica, podem retornar duas respostas possíveis, positiva quando se reconhece o acesso e negativa quando se nega o acesso de uma pessoa frente à comparação de sua característica biométrica. A partir das duas respostas possíveis é possível encontrar quatro situações (MORAES, 2010), representadas pela Figura 6, são elas:

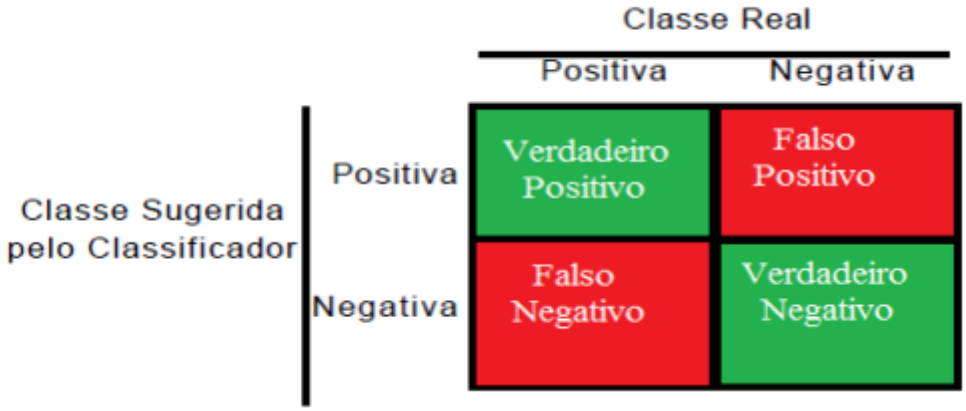
1. Verdadeiro Positivo – *True Positive*: O padrão encontrado é verdadeiro e o sistema identifica-o como tal;

2. Verdadeiro Negativo – *True Negative*: O padrão encontrado é falso e o sistema identifica-o como tal, negando o acesso;

3. Falso Positivo – *False Positive*: O padrão é falso, porém o sistema o reconhece como verdadeiro, autenticando uma pessoa de maneira errada;

4. Falso Negativo – *False Negative*: O padrão é verdadeiro, porém o sistema o identifica como negativo, negando a sua autenticidade.

Figura 6 – Tabela de contingência com as quatro situações possíveis para um sistema binário.



Fonte: Moraes (2010)

2.2.1 Biometria facial

Identificar pessoas é muito simples para os seres humanos, porém é um problema muito complexo computacionalmente e vem sendo pesquisado durante muitos anos para chegar no nível tecnológico que possuímos hoje. Para Tolba, El-Baz e El-Harby (2008) o incrível avanço do reconhecimento facial se deve a combinação de dois fatores, são eles: desenvolvimento contínuo e ativo de algoritmos, disponibilidade de grandes bases de dados de imagens faciais e métodos de validação para verificar o desempenho de algoritmos de reconhecimento facial.

Segundo Moraes (2010), as soluções mais populares que resolvem o problema de reconhecimento facial, baseiam-se na localização e análise de atributos faciais como nariz, olhos, ou da análise geral da face.

Uma das grandes vantagens do reconhecimento facial se deve ao fato de ser um método não intrusivo (TOLBA; EL-BAZ; EL-HARBY, 2008), de fácil coleta, alta aceitação e universalidade (MORAES, 2008).

O fluxo básico de sistemas de biometria facial possui quatro etapas básicas (MORAES, 2008), são elas:

1. Detecção da existência de face em imagem ou em vídeo;

2. Localização da face na imagem;

3. Extração das características faciais (posição nariz, boca, olhos) e comparação com base de conhecimento;

4. Retorno do padrão armazenado mais próximo do padrão de assinatura biométrico da imagem de entrada.

Um dos processos mais importantes do reconhecimento facial para aumentar a taxa de asserção está no processo de normalização. Fatores como alterações na face, como por exemplo, a presença ou ausência de barba, prejudicam o reconhecimento facial.

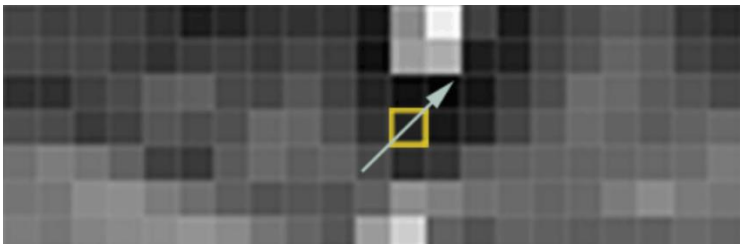
As tecnologias de reconhecimento facial ainda possuem o desafio de resolver problemas como iluminação e posição da face em sentido de rotação do rosto. Por situações como estas, sistemas de biometria facial ainda possuem desempenho relativamente baixo se comparados com sistemas de impressão digital e íris (TOLBA; EL-BAZ; EL-HARBY, 2008).

2.3 Reconhecimento facial

Dentre as diferentes propostas para reconhecimento de faces em imagens, o algoritmo baseado em Histogram of Oriented Gradients (HOG), destaca-se por ser capaz de extrair características de imagens por meio da diferença da orientação dos gradientes existentes (DALAL; TRIGGS, 2005).

A detecção facial com a utilização de algoritmo em HOG inicia com a conversão da imagem para preto e branco, pois não é necessário o maior detalhamento de cores para este algoritmo. Para cada *pixel* da imagem é realizado uma comparação com os *pixels* que o cercam, identificando o quanto o *pixel* atual é mais escuro quanto aos *pixels* a sua volta. Neste processo são definidas e desenhadas linhas no sentido em que a imagem se torna mais escura, conforme Figura 7 (GEITGEY, 2016).

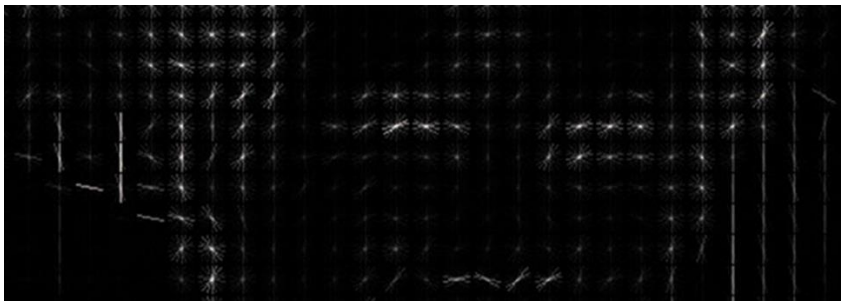
Figura 7 – Comparação dos gradientes onde a imagem se torna mais escura



Fonte: Geitgey (2016)

Este processo de criação de linhas resulta em um gradiente da imagem que mostra todo o fluxo do branco para o preto. Este processo visa resolver a complexidade de detectar faces, porque até mesmo a face de uma pessoa apresenta valores completamente diferentes se comparada com imagens escuras e imagens mais claras. Para resolver o grande detalhamento deste gradiente a imagem é novamente quebrada em blocos de 16x16 *pixels*, realizando um processo similar ao anterior, que define o principal sentido dos gradientes dentro do bloco, resultado evidente conforme Figura 8 (GEITGEY, 2016).

Figura 8 – Imagem original é convertida para uma representação em HOG



Fonte: Geitgey (2016)

Com a imagem convertida em HOG é realizado a comparação com um padrão HOG criado através de inúmeras faces, assim se houver uma face na imagem com um padrão similar ao padrão HOG facial, a face será definida como detectada (GEITGEY, 2016).

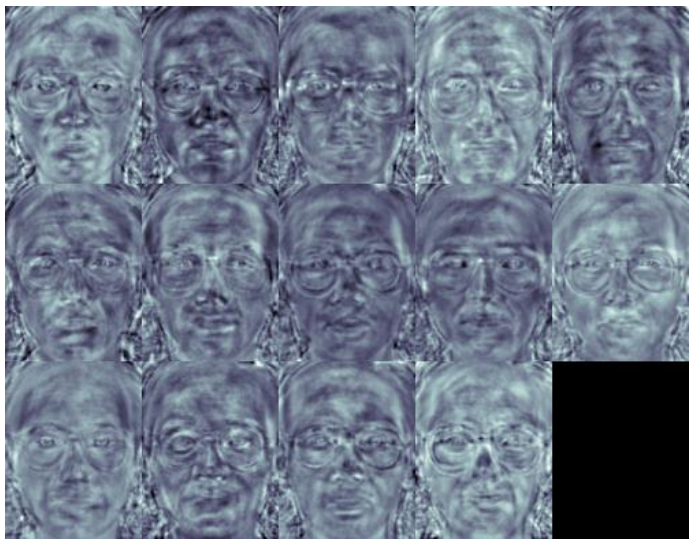
2.4 Biometria facial com Fisherfaces

O algoritmo para identificação biométrica facial de pessoas Fisherfaces, disponibilizado pelo OpenCV, é baseado em algoritmo de classificação inicialmente proposto para a classificação de flores em 1936, realizando uma análise conhecida como Linear Discriminant Analysis (LDA) que busca maximizar a dispersão entre as classes e minimizar a dispersão dentro de cada classe. Este algoritmo segue uma ideia simples, deixar classes iguais mais próximas enquanto classes diferentes são afastadas (OPENCV, 2019).

O algoritmo não sofre grande influência das condições de luminosidades existentes nas imagens, se comparado com o algoritmo Eigenfaces, também disponibilizado pelo OpenCV. Ele é capaz de criar uma matriz de transformação com base em uma classe específica, a LDA encontra as características faciais que diferenciam as pessoas (OPENCV, 2019).

Conforme Figura 9 é possível visualizar o resultado da aplicação do algoritmo Fisherfaces para identificação facial de pessoas.

Figura 9 – Resultado do algoritmo Fisherfaces



Fonte: OPENCV (2019)

Este algoritmo foi testado durante o desenvolvimento do protótipo para validar o reconhecimento facial dos usuários.

2.5 Machine Learning

Para a computação, Machine Learning (aprendizagem de máquina) representa a capacidade de com diferentes algoritmos, seja realizado a detecção automática de padrões significativos em dados, de maneira que um sistema possa inferir respostas ou tomar decisões que automatizem processos. Largamente utilizada nas últimas décadas em qualquer tarefa que exija a extração de informação em grandes quantidades de dados. Sua aplicabilidade é imensa, sendo encontrada desde em sistemas de e-mail para nos proteger de *spams* até a identificação de transações fraudulentas de cartão de crédito.

Machine Learning se faz interessante em resolver problemas em que dependendo da complexidade do padrão a ser detectado, um humano não conseguiria definir uma regra ou um conjunto explícito de regras que poderiam ser utilizadas na classificação dos dados, porém com a utilização de diferentes algoritmos, a máquina é capaz de aprender a relação dos dados e padrões existentes e assim passe a inferir respostas (SHALEV-SCHWARTZ; BEM-DAVID, 2014).

2.5.1 Tipos de aprendizagem

Machine Learning pode ocorrer de diferentes maneiras, dependendo do problema a ser resolvido, destacando-se três modos principais: o modelo supervisionado, o modelo não supervisionado e a aprendizagem por reforço.

No modelo supervisionado, a máquina aprende com base em um conjunto de dados de exemplo, onde que há uma definição ou marcação do que seria a resposta correta para o conjunto de dados. Assim posteriormente a máquina poderia inferir uma resposta quando novos dados fossem colocados à prova. Este modelo é similar a quando um professor ensina seu aluno a memorizar com base em bons exemplos, criando uma noção geral de regras baseadas nos exemplos fornecidos (MUELLER; MASSARON, 2016).

Diferente do modelo supervisionado que necessita de modelos pré-definidos de resposta, no modelo não supervisionado, os padrões são encontrados sem qualquer interferência. São conhecidos também como modelos exploratórios, por encontrarem padrões

que são desconhecidos em um conjunto de dados (SIDEY-GIBBONS; SIDEY-GIBBONS, 2019).

Este modelo é muito útil para dar ideias e encontrar algum significado em conjunto de dados onde que os padrões são desconhecidos, útil também por prover conjunto de novos valores a serem utilizados no modelo de aprendizagem supervisionado. Este modelo também costuma reestruturar o conjunto de dados de entrada em algo novo, identificando novas características que podem ser usadas para criar uma classificação ou simplesmente uma série de valores não relacionados (MUELLER; MASSARON, 2016). Como aplicação cita-se modelos de recomendação de filmes, que procuram sugestões ao relacionar gostos de entre pessoas diferentes.

A aprendizagem por reforço ocorre de maneira semelhante ao modelo não supervisionado, pois não há uma resposta definida para o conjunto de dados, porém neste modelo acompanhamos as soluções propostas pela máquina e definimos um *feedback* positivo ou negativo, assim cada decisão tem uma consequência, compara-se ao modelo de aprendizagem humana, na tentativa e erro. De maneira que cada situação de *feedback* negativo a máquina aprende que não deveria seguir determinado caminho, como quando poderia haver maior custo ou tempo investido (MUELLER; MASSARON, 2016).

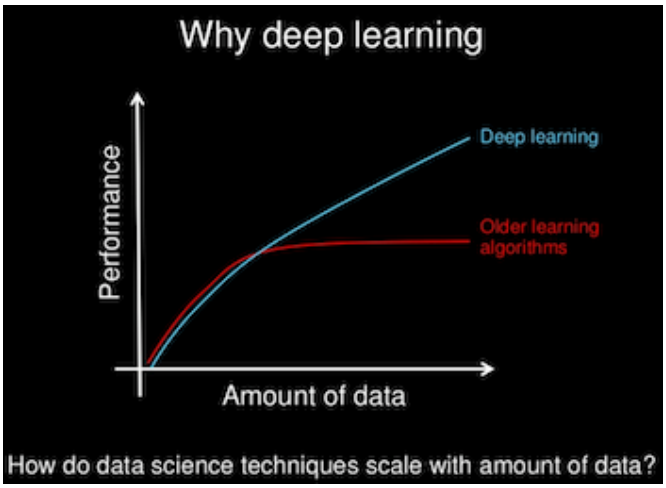
2.5.2 Deep Learning

Deep Learning é uma subseção de Machine Learning, focada em algoritmos inspirados na estrutura e funcionamento do cérebro, denominados de redes neurais artificiais. As redes neurais artificiais são construídas de modo similar a um cérebro humano, onde há nós de neurônios que se conectam como uma rede, de maneira hierárquica, o processamento de uma camada de neurônios serve como fonte de entrada para a camada posterior, a cada avanço as informações se tornam mais especializadas. Este modelo de estrutura possibilita que a aprendizagem ocorre de maneira não linear, diferente da maneira tradicional do funcionamento de algoritmos baseados em Machine Learning (BROWNLEE, 2019).

Para o treinamento em grandes conjuntos de dados o modelo tradicional de Machine Learning não se torna eficiente, conforme ilustrado na Figura 10, para o modelo tradicional de

aprendizagem há um problema de como escalar a aprendizagem, cada vez mais informações surgem e o desempenho não aumenta conforme mais informações são inseridas, em contrapartida Deep Learning possibilita a escalabilidade do processamento (NG, 2015).

Figura 10 – Escalabilidade em Machine Learning



Fonte: Ng (2015)

Algoritmos baseados em Deep Learning podem ser aplicados para inúmeras áreas, como visão computacional, reconhecimento de imagens, sistemas de reconhecimento de voz, reconhecimento de padrões (AJIT, 2015).

O processamento de imagens é altamente sensível e dependente da localização das características que existem em uma região, para que se possa inferir e classificar informações em uma imagem (BROWNLEE, 2019). Para abordar este problema pode ser utilizado Convolutional Neural Network (CNN), um algoritmo capaz de representar as características que existem em uma imagem, atribuindo importância à vários elementos que existem nela e também sendo capaz de diferenciar uns dos outros. O pré-processamento necessário em uma *convolutional network* é muito menor se comparado com outros métodos de classificação, este algoritmo é capaz de aprender a classificar e filtrar as características se corretamente treinado (SAHA, 2018).

2.6 Internet das coisas

A União Internacional de Telecomunicações (*International Telecommunication Union* – ITU), define Internet of Things (IoT), como uma infraestrutura global de informação que possibilita serviços avançados ao conseguir conectar diferentes dispositivos tecnológicos através de comunicação interoperável (ITU, 2012).

O termo “Internet das coisas” foi utilizado pela primeira vez em 1991 por Kevin Ashton ao comentar do uso de etiquetas com tecnologia Radio Frequency Identification (RFID) e como isso poderia revolucionar a vida das pessoas. Este conceito representa a capacidade de comunicação entre dispositivos por meio da Internet, possibilitando a criação de soluções de diferentes focos, tanto industrial, quanto doméstico (GOMES; BERGAMO, 2018).

A IoT possibilita que dispositivos de uso comum das pessoas possuam novas funcionalidades e automações que não seriam possíveis sem a maior conectividade que estas tecnologias trazem. Conforme Gomes e Bergamo (2018), o relógio que apenas servia para mostrar as horas, agora pode receber ligações, e-mails e monitorar os dados sobre a saúde de seu usuário. A geladeira que servia unicamente para armazenar os alimentos, agora pode controlar prazos de validade e até mesmo realizar compra de produtos que terminaram.

Nunes (2016), explica que um dispositivo eletrônico ou eletromecânico é dito como “inteligente” (*smart*) quando pode operar de maneira autônoma e interagir com outros dispositivos, possuindo capacidade de processamento, armazenamento de dados e comunicação sem fio e componentes físicos necessários para a realização de suas funções.

Conforme Nunes (2016), faz parte do IoT, simples dispositivos inteligentes denominados de sensores que medem variáveis do ambiente como temperatura e umidade, realizando a transmissão destes dados para dispositivos controladores, que podem ser computadores ou smartphones ou outros dispositivos conectados à Internet. Os sensores que possuem alguma lógica frente aos valores lidos denominam-se como atuadores, como exemplo um sensor de temperatura que controla o acionamento de um aquecedor quando a temperatura fica abaixo do programado, o diferente neste contexto é a conectividade deste atuador que quando ligado à Internet pode ser controlado à distância.

No contexto de “casas inteligentes”, Nunes (2016), define como tal quando há vários tipos de sensores e atuadores instalados, usados para controlar de maneira autônoma, serviços dentro de casa, conforme exemplificado pela Figura 11, como controle de temperatura, luminosidade.

Figura 11 – Representando diferentes dispositivos conectados na casa inteligente.

Fonte: Neocontrol (2019)

2.7 Casas inteligentes

Conforme Benyon (2014), existem princípios gerais de design para casas inteligentes, que definem expectativas e meios de interação que são esperados, são alguns deles:

- As pessoas vivem com muita pressa e normalmente não estão preocupadas em realizar tarefas do cotidiano, indicando a importância e a velocidade com que sistemas de interação precisam se adaptar, se encaixando no modelo de vida e padrão de pessoas;
- Pessoas buscam criar meios ideais para a sua melhor experiência em casa;
- Pessoas buscam tecnologias que se integram e façam parte do ambiente, as interfaces devem se tornar transparentes e focar não em apenas funcionalidades, mas em experiências;
- A interação com a casa deve ser fácil e mais natural;

- A casa deve respeitar as preferências de cada pessoa;
- A casa deve se adaptar às questões físicas e sociais, como exemplo, o perfil de configuração da TV deve ser diferente na presença de um grupo de pessoas de quando há apenas uma;
- A casa deve antecipar a necessidade e desejo das pessoas;
- A interação com a casa inteligente deve ser confiável, as aplicações devem levar em consideração questões de privacidade.

2.8 Consumo de banda para utilização de múltiplas câmeras

Por propor a utilização de múltiplas câmeras em um contexto de casas inteligentes, se faz necessário a compreensão de um consumo de banda estimado para transmissão de imagens em tempo real e contínuo. Porém estimar este valor não é tão simples, diversos fatores podem influenciar no tamanho das imagens geradas, estes fatores se dividem em controlados e não controlados. Os fatores que podem ser controlados são resolução, *frames* por segundo e compactação da imagem. Os fatores que não podem ser controlados são quantidade de movimento e quantidade de informação contida nas imagens (YOUSEG, 2017).

Para a realização de uma estimativa é necessário que os fatores de controle sejam definidos, a configuração é definida conforme:

- Compressão: O formato compressão comumente utilizado em câmeras de monitoramento é o MJPEG, este formato é constituído de uma sequência de imagens compactadas Joint Photographic Experts Group (JPEG) (CISCO, 2019).
- *Frames* por segundo: A quantidade de *frames* por segundo varia de acordo com a necessidade, para situações de monitoramento e gravação não são necessários valores acima de 15fps (CISCO, 2019).

- Resolução: Não é necessário a utilização de grandes resoluções para a proposta deste sistema, para facilitar a estimativa será definido uma resolução média de 1280x800px.

O resultado da estimativa é realizado com base na multiplicação dos três (resolução x fps x compactação). Para uma imagem de 1 megapixel com resolução 1280x800px com compactação MJPEG o tráfego estimado é de 0.95 Mbps (YOUSEG, 2019).

Se forem utilizadas 4 câmeras conectadas ao sistema proposto, com uma taxa de 15fps a largura de banda estimada será de 54 Mbps (0.95 x 15 x 4).

2.9 Trabalhos relacionados

Nesta seção serão apresentados outros trabalhos de pesquisa que possuem relação com a proposta tratada neste trabalho. O estudo destes trabalhos contribuiu como base tecnológica quanto as possibilidades existentes para reconhecimento de gestos estáticos, biometria facial e interação com casas inteligentes, além de contribuir no levantamento de requisitos para o desenvolvimento deste projeto.

2.9.1 Biometria facial com OpenCV

No trabalho realizado por Galimberti (2018), o autor realizou um estudo comparativo dos algoritmos para reconhecimento facial disponibilizados pelo OpenCV para validação de um sistema que realiza o controle de acesso com base em biometria facial para autenticação.

Para o estudo comparativo foram analisados os algoritmos Local Binary Patterns Histograms, Eigenfaces e Fisherfaces que apresentam diferentes soluções para a extração da biometria facial. O autor realizou o treinamento com base nestes diferentes algoritmos e validou os resultados com diferentes amostras. Conforme a Figura 12 é possível verificar o teste realizado para uma amostra de imagens coletadas no decorrer de duas semanas, com diferentes luminosidades e condições ambientais.

Figura 12 – Resultados do teste de uma amostra

	LPBH	Eigenfaces	Fisherfaces
Verdadeiros Positivos	8	4	10
Verdadeiros Negativos	42	45	45
Falsos Positivos	4	1	0
Falsos Negativos	51	55	50
Taxa de Falsa Rejeição	86,00%	93,00%	83,00%
Taxa de Falsa Aceitação	8,00%	2,00%	0,00%

Fonte: Galimberti (2018)

É possível verificar o melhor resultado para o algoritmo Fisherfaces, além de que todos os algoritmos apresentam uma alta taxa de falsa rejeição, devido a alta rigorosidade necessária para sistemas de controle de acesso.

O autor concluiu o trabalho apresentando dentre os algoritmos do OpenCV, o Fisherfaces, como o de melhor resultado frente aos demais, tanto em velocidade quanto na taxa de reconhecimento. O autor ainda apontou de que melhores resultados podem ser obtidos se os algoritmos de reconhecimento forem utilizados em conjunto.

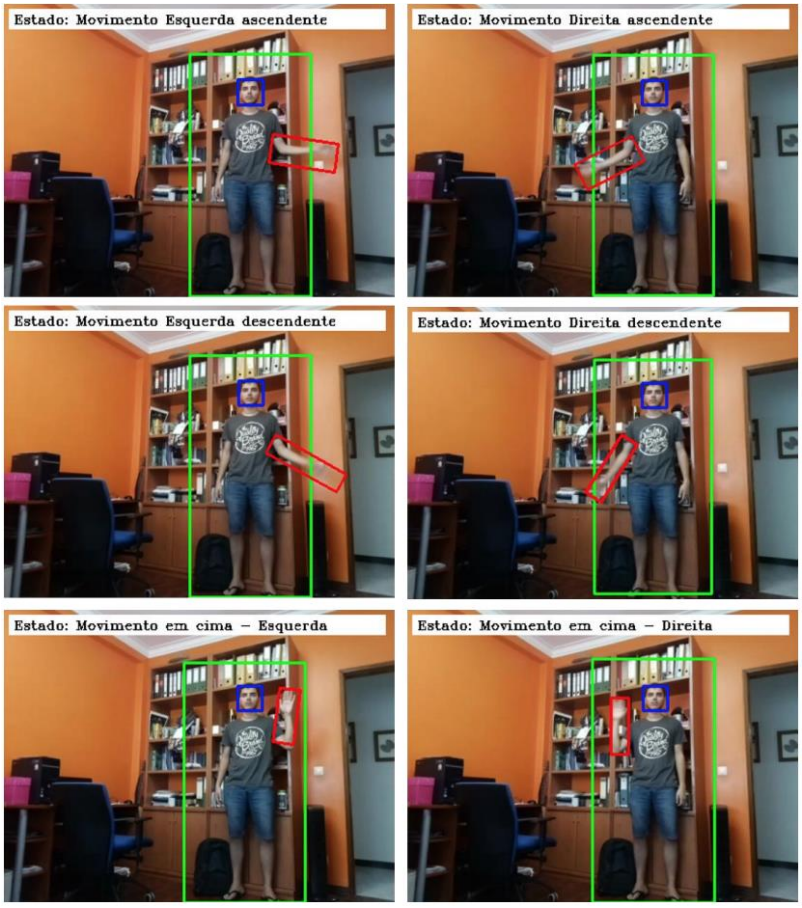
O trabalho de Galimberti (2018) se encontra com este no sentido de validar o controle de acesso à algum recurso por meio da identificação da biometria facial dos usuários, assim foi utilizado o algoritmo de melhor reconhecimento identificado no projeto de Galimberti (2018), o Fisherfaces, como proposta para teste deste trabalho, além de explorar a utilização de modelo de reconhecimento facial baseado em Deep Learning para identificação dos usuários. Os resultados da comparação destes algoritmos serão vistos na seção de resultados deste trabalho.

2.9.2 Reconhecimento de gestos para interação com robô

No trabalho realizado por Gonçalves (2017), o autor desenvolveu uma aplicação para reconhecimento de gestos baseado em visão computacional que é integrado a um robô já existente para que crianças que se encontram em tratamento hospitalar possam brincar e interagir, criando uma atividade lúdica e de distração para a situação em que elas se encontram.

O projeto foi desenvolvido de maneira embarcada em um Raspberry Pi, fazendo o uso de uma câmera conectada ao hardware é realizado a captura e o processamento dos *frames*. A aplicação foi desenvolvida com o *framework* OpenCV tanto para captura quanto o processamento das imagens, sendo utilizado os classificadores em cascata disponibilizados pelo OpenCV baseados em Histogram Oriented Gradients (HOG). Assim que uma pessoa é detectada é iniciado a detecção do movimento das mãos. Conforme a Figura 13, aplicação realiza a identificação do movimento dos membros superiores, como braço ascendente, descendente, à direita ou à esquerda e para cada movimento é emitido um som correspondente.

Figura 13 – Visualização de todos os movimentos reconhecidos pela aplicação



Fonte: Gonçalves (2017)

Gonçalves (2017) comenta ainda sobre o requisito mínimo e máximo de distância da pessoa frente a câmera para o correto funcionamento da aplicação, onde que tanto para distâncias superiores a 4 metros ou inferiores a 1.2 metros não é possível reconhecer o sujeito da interação. As conclusões do autor foram negativas para a capacidade de processamento em

tempo real para altas resoluções, assim para média e baixa resolução há um processamento adequado.

Para o presente trabalho, também houve a necessidade de explorar diferentes configurações durante o desenvolvimento para obtenção de melhores resultados quanto ao desempenho do protótipo.

O projeto de Gonçalves (2017), diferente do projeto desenvolvido neste trabalho, realiza a detecção de movimentos dos membros superiores do corpo, detectando a diferença da posição dos membros superiores entre os *frames* para assim definir o sentido de movimento. O protótipo desenvolvido neste trabalho realiza a detecção estática de gestos das mãos realizada a cada *frame* com base em uma modelo de reconhecimento previamente treinado.

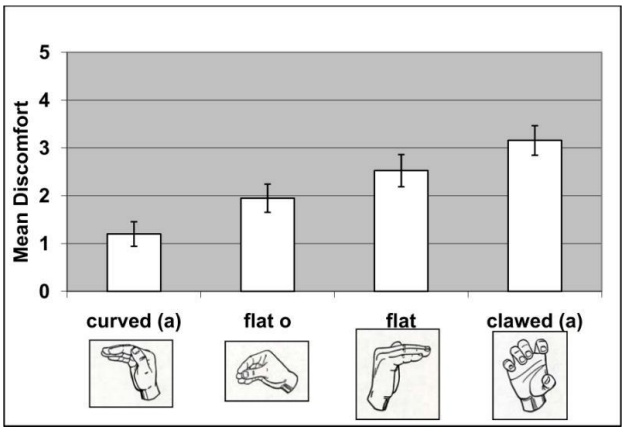
2.9.3 Estudo sobre design da utilização de gestos para interação Humano Computador

O trabalho de Rempel, Camilleri e Lee (2015) realizou um estudo sobre a interação humano computador com o uso de gestos e o impacto desta para o processamento cognitivo e fadiga resultante da interação. Os autores buscaram criar uma orientação com base em informações de intérpretes que pudesse definir uma interação mais natural, com menos dor e fadiga.

O estudo foi realizado com 24 intérpretes experientes em linguagem de sinais, aplicando um questionário para avaliar o nível de desconforto associado às diferentes posturas, movimentos e caracteres das mãos usados durante a realização dos sinais. Os níveis de desconforto foram avaliados quanto à localização das mãos, posturas e movimento das mãos e posturas dos dedos. O local de maior conforto detectado para as mãos durante a realização dos gestos foi na altura do peito, região inferior e próximo ao corpo. As posturas e movimentos de maior conforto foram que envolviam o punho reto ou neutro, com as palmas voltadas uma para a outra ou giradas levemente em direção ao chão. Já rotações extremas dos punhos, como superiores a 45 graus geram desconforto. Para os dedos a melhor postura foi com todos os dedos levemente flexionados, sendo menos confortável a realização de movimentos com os dedos esticados ou afastados.

Conforme Figura 14, é possível verificar parte do estudo, onde é possível verificar o aumento de desconforto conforme o gesto realizado exige que a mão seja flexionada.

Figura 14 – Gráfico de desconforto na realização de gestos



Fonte: Rempel, Camilleri e Lee (2015)

Rempel, Camilleri e Lee (2015) identificaram por meio de seu trabalho que a maioria dos intérpretes de linguagem de sinais sentem dor, desconforto e fadiga nas mãos e braços, os sintomas associados as dores não são triviais, causam dores prolongadas. E a interação humano computador pode apresentar o mesmo risco se realizado durante muitas horas por semana. Os autores ainda ressaltaram que a utilização de gestos pode substituir parcialmente ou toda utilização de teclado e mouse como fonte de entrada, podendo ser realizado de 20 a 50 horas por semana, valor muito superior ao tempo de que intérpretes realizam durante seus trabalhos, se configurada corretamente.

Rempel, Camilleri e Lee (2015) concluíram a pesquisa com importância da escolha para os gestos que são mais usados para a interação desejada, buscando criar uma interação natural e confortável para os usuários, além da utilização de gestos que sejam simples cognitivamente para os usuários e que não demandam grande tempo para a sua realização.

Para a interação com protótipo desenvolvido o tempo necessário é pequeno e a realização do gesto pode ocorrer conforme orientação de Rempel, Camilleri e Lee (2015),

realizando gestos na altura do peito, com gestos onde o punho está reto e os dedos próximos um ao outro, sem extensão completa.

2.9.4 Desenvolvimento de uma Abordagem para o Reconhecimento de Gestos Manuais Dinâmicos e Estáticos

No trabalho realizado Cárdenas (2015), foi realizado o desenvolvimento de uma abordagem para diferentes tipos de gestos manuais, tanto estáticos quanto dinâmicos, tendo como objetivo a melhora da interação humano computador. O trabalho do autor foi realizado com a utilização do sensor Kinect que além de informações como intensidade, profundidade e posições das articulações do corpo, ainda realiza segmentação e detecção da mão.

Para a criação do modelo de reconhecimento de gestos estáticos foi utilizado a informação de profundidade para propor um método baseado na representação da mão como uma nuvem de pontos. Com base em uma teoria de cossenos de direção foi criado um Histograma de Magnitudes Acumuladas para representação das características da mão a serem utilizadas em um classificador baseado em Support Vector Machine (SVM).

Conforme Cárdenas (2015), seu modelo de classificação de gestos estáticos gerou resultados superiores a outros modelos da literatura, com melhor resultado de 99.21% de acurácia média.

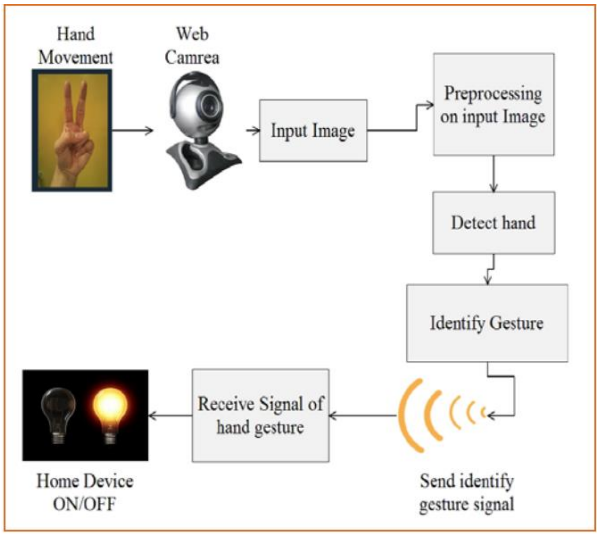
O protótipo desenvolvido neste trabalho, diferente da proposta de Cárdenas (2015), realiza por meio de diferentes algoritmos e classificadores do OpenCV a detecção da mão e segmentação, não havendo a informação como profundidade que poderia possibilitar o aumento da qualidade dos resultados obtidos. Para o desenvolvimento do protótipo deste projeto foi feito o uso de uma câmera integrada mais simples que não fornece tais informações.

2.9.5 Dispositivo de controle doméstico com base no reconhecimento de gestos

No trabalho realizado Patel, Pandya e Patel (2017), foi desenvolvido um sistema para automação do controle de dispositivos IoT domésticos, auxiliando pessoas com deficiência

física a controlar estes dispositivos por meio do reconhecimento de gestos. Conforme Figura 15, com base na captura de imagens de uma câmera é realizado a segmentação do fundo da imagem e então a detecção da mão para realização da identificação do gesto com base na contagem de dedos. O gesto reconhecido é convertido em um valor binário e através da rede é enviado um sinal de comunicação para um hardware modular Raspberry Pi realizar o controle com o dispositivo correspondente, conforme sinal recebido.

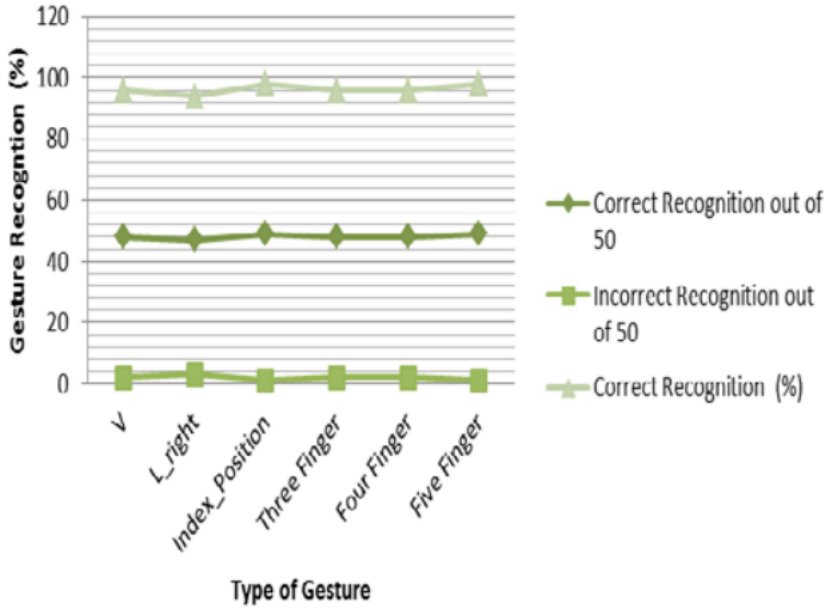
Figura 15 – Proposta para controle de dispositivos IoT



Fonte: Patel, Pandya e Patel (2017)

No trabalho de Patel, Pandya e Patel (2017) não foi especificado com clareza a forma de todos os gestos que o sistema é capaz de reconhecer, para o reconhecimento dos gestos estáticos não foi feito o uso de Machine Learning, puramente processamento visual das imagens. Os autores testaram apenas o reconhecimento dos gestos, não houve detalhamento do desempenho da solução, os testes de reconhecimento realizados estão visíveis na Figura 16. Conforme resultados é possível verificar que os gestos foram reconhecidos corretamente com uma acurácia de praticamente 100% para todos os gestos, porém não há maiores detalhes sobre a implementação da solução e configuração utilizada para realização de uma análise mais completa.

Figura 16 – Resultados do reconhecimento de gestos



Fonte: Patel, Pandya e Patel (2017)

Patel, Pandya e Patel (2017) concluíram que tecnologias IoT também podem ser utilizadas para o desenvolvimento de soluções em casas inteligentes, fornecendo conforto para melhorar a qualidade de vida das pessoas. O controle de gestos também é uma forma possível para automação doméstica, fornecendo uma linguagem mais natural para comunicação humano computador.

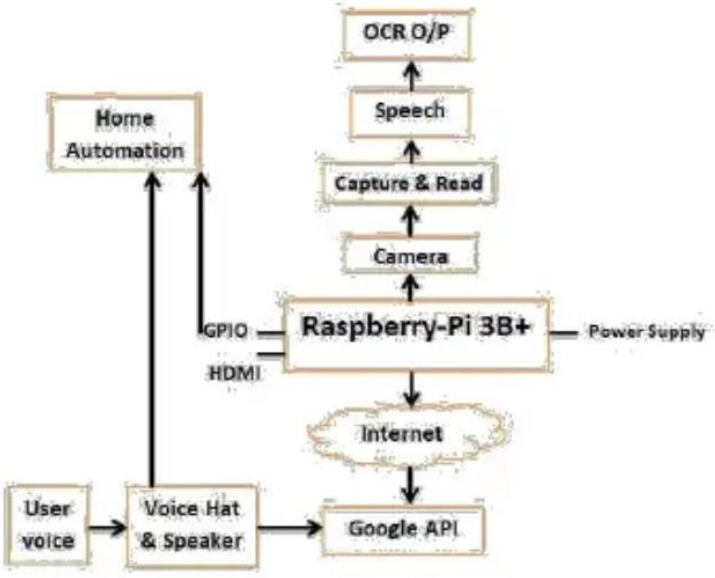
2.9.6 Um sistema assistencial para pessoas deficientes visuais usando Raspberry Pi

Dubey, Verma e Mehendale (2019), desenvolveram um sistema assistencial que combina leitura automatizada de documentos (OCR) e assistente pessoal virtual para automação doméstica, focada em pessoas com deficiência visual. Para o desenvolvimento da assistente pessoal foi utilizado a assistente pessoal Google que prove diferentes funções para automação de diferentes atividades cotidianas como verificação de e-mails, clima e noticiais.

O sistema foi desenvolvido em Python, de maneira embarcada em um Raspberry Pi, juntamente com uma câmera (Pi-Camera) integrada. O diagrama de blocos da solução proposta pode ser visualizado na Figura 17, onde é possível verificar que mediante a captura da voz, é

realizado o processamento mediante comunicação com a API do Google para processamento da linguagem e conforme o retorno desta comunicação é realizado a automação no ambiente doméstico. O sistema também possibilita que mediante a apresentação de algum documento de texto em frente à câmera seja realizado a leitura do conteúdo, este processo ocorre com a utilização de um *framework* OCR juntamente com a tecnologia para conversão de texto em áudio do Google, o Google Text to Speech Engine.

Figura 17 – Diagrama de blocos



Fonte: Dubey, Verma e Mehendale (2019)

Os autores Dubey, Verma e Mehendale (2019) concluíram o trabalho demonstrando interesse para inclusão de novas funcionalidades para o projeto, como controle de segurança, aviso de obstáculos. Ainda ressaltaram que assistentes pessoais, como Amazon Echo, falharam em fornecer um senso de controle ao usuário, pois algumas vezes não retornam respostas aos comandos inválidos de voz realizados e também possuem alto custo para aquisição.

3 METODOLOGIA

O presente trabalho apresenta o desenvolvimento de um protótipo de solução que permite a interação com dispositivos IoT por meio da identificação da biometria facial e reconhecimento dos gestos estáticos. Para o desenvolvimento do sistema proposto diferentes estratégias foram pensadas para criar uma solução viável, diferentes tecnologias foram exploradas e testadas ao decorrer do processo, desta forma, o presente trabalho se caracteriza como uma pesquisa exploratória, que conforme Gil (2006), busca criar uma maior base de conhecimento sobre o problema, incentivando a criação de hipóteses e soluções.

Gil (2006) cita que pesquisas exploratórias são bastante flexíveis, possibilitando um escopo dinâmico, conforme a necessidade. A pesquisa pode ser composta por uma revisão bibliográfica, entrevistas com pessoas que tiveram experiências relacionadas e análise de exemplos que clarifiquem a compreensão do problema.

A partir do estudo bibliográfico e a análise do estado atual dos algoritmos de visão computacional foi especificado o projeto e seus requisitos. Com base na definição do projeto foi realizado desenvolvimento de um protótipo no qual foram realizados testes a fim de validar a viabilidade desta solução, foram analisados os resultados com base nos seguintes critérios: identificação do gesto, assertividade da biometria facial e o tempo de resposta final.

3.1 Tecnologias utilizadas

Visando o desenvolvimento do protótipo do sistema proposto neste trabalho foram utilizadas as tecnologias descritas nas subseções a seguir.

3.1.1 Python

Conforme Müller e Guido (2017), Python se tornou a linguagem em comum entre muitas aplicações focadas em ciência de dados. Python possui bibliotecas para carregamento, visualização, estatística, processamento de imagens.

Esta linguagem foi escolhida devido a vasta quantidade de projetos e tecnologias voltadas para a área de visão computacional.

3.1.2 OpenCV

OpenCV é o principal projeto de código aberto para a visão computacional, processamento de imagem e Machine Learning, com suporte a processamento na GPU (NVIDIA, 2019).

Devido ao suporte dos variados tipos de algoritmos para processamento de imagens, o OpenCV foi um *framework* de suma importância para a resolução dos problemas de visão computacional deste trabalho.

3.1.3 NodeJS

O NodeJS é um ambiente de execução Javascript multi-plataforma, construído em cima da *engine* do Chrome, o V8. O NodeJS utiliza de um modelo orientado a eventos, não

bloqueantes que o torna leve e rápido, com foco em aplicações *server-side* e de rede (TURNKEY LINUX, 2019).

Esta tecnologia foi utilizada para o desenvolvimento de uma API REST que oferece uma camada de comunicação para o controle dos dispositivos IoT conectados na placa de prototipação Arduino Uno.

3.1.4 Arduino

Arduino é uma plataforma de eletrônica *open-source* baseado em um sistema e hardware de fácil utilização, podendo ser utilizado para o desenvolvimento de diferentes tipos de aplicações, desde o controle de um LED até o controle de um motor (ARDUINO, 2019).

O Arduino Uno utilizado neste trabalho é visível na Figura 18 e possui a seguinte especificação:

- Microcontrolador: ATmega328;
- Tensão de operação: 5V;
- Pinos digitais I/O: 14;
- Pinos de entrada analógica: 6.

Figura 18 – Arduino Uno

Fonte: Arduino (2019)

3.1.5 Johnny-Five

O projeto Johnny-Five é um *framework de* robótica Javascript e plataforma IoT, disponibilizando uma camada única para comunicação com diversos hardwares modulares (JOHNNY-FIVE, 2019).

Este projeto foi utilizado juntamente com a API REST em NodeJS, realizando o controle do hardware modular Arduino Uno.

4 PROJETO

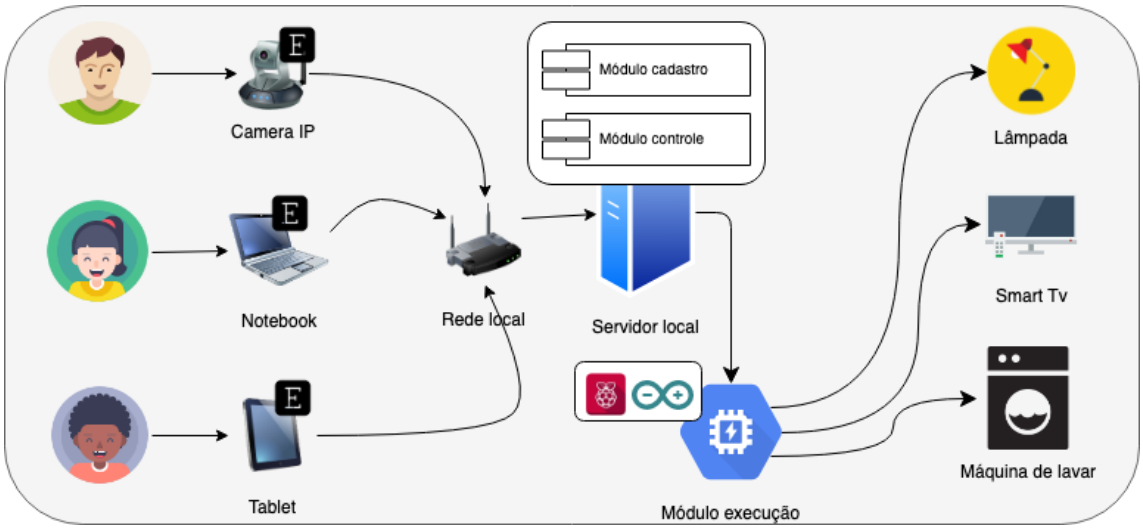
Neste capítulo são abordados os detalhes sobre cada um dos módulos especificados para resolver o problema estudado, assim como cada um interage com o todo.

Para a criação de um sistema modularizado e que segue padrões e boas práticas de programação, foram necessários a criação de três módulos, são eles: Cadastro, Controle e Execução.

A Figura 19 apresenta o fluxo de interação dos módulos e atores da arquitetura. Diferentes pessoas dentro de um contexto compartilhado, como em uma casa inteligente, se conectam e interagem com a aplicação através de fontes de entrada de imagem e vídeo, assim representados pela letra E (Entrada). Estes dispositivos de entrada quando conectados são consumidores do serviço, assim reconhecidos como clientes do sistema. Cada cliente é conectado através da rede com o módulo Controle que realiza o processamento de visão computacional, assim como a identificação facial e o reconhecimento de gestos estáticos. Ao identificar uma pessoa autorizada e o gesto realizado pela mesma, é disparado uma comunicação com o módulo Execução que se encarrega de executar a ação correspondente com os diferentes dispositivos conectados, como lâmpadas, máquinas de lavar e, *smart* TV.

No servidor local são disponibilizados e concentrados o módulo de Cadastro e o módulo Controle, conforme Figura 19, demonstrando assim que o processamento de imagem ocorrerá localmente, assim como questões administrativas da solução. Para o módulo Execução foi estipulado o uso de hardwares modulares como Arduino Uno ou Raspberry Pi, como meio para intermediar comunicações com diferentes dispositivos que possam estar conectados. O uso destes hardwares é interessante devido ao seu baixo custo e consumo de energia, além de possuírem saídas digitais e analógicas de fácil controle.

Figura 19 - Fluxo de interação e comunicação do projeto.



Fonte: Elaborado pelo autor (2019).

Por realizar processamentos mais complexos, como biometria facial e reconhecimento de gestos estáticos, esta etapa de processamento é realizada no módulo Controle, que é disponibilizado dentro de um servidor local. Assim sendo possível garantir menor tempo de resposta e evitando problemas de latência e disponibilidade. Desta forma, propõe-se o desenvolvimento de um modelo conhecido como Edge Computing, justamente por manter questões administrativas e de processamento locais, sem consumo de algum serviço em *cloud*. Edge Computing se faz necessário por aproximar o processamento para o local de utilização, resolvendo problemas de latência e disponibilidade. Conforme Shi, Li, Xu e Zhang (2016) provas de conceito indicaram redução do tempo de resposta de 900ms para 169ms em aplicações de reconhecimento facial, ao realizar a migração de modelos de Cloud Computing para Edge Computing.

Por estar hospedado dentro de um servidor local, há também a possibilidade de que múltiplas fontes de entrada como imagem e vídeo sejam facilmente adicionadas, possibilitando fontes de entrada de vídeo de diferentes tipos, como câmeras de monitoramento e computadores pessoais. Assim cada fonte de entrada que possui uma câmera passa a assumir o papel de cliente do serviço, quando que conectada ao módulo Controle. Porém é importante ressaltar que a qualidade da câmera de cada fonte de entrada impacta diretamente a capacidade e a qualidade dos resultados que são obtidos, a identificação facial e de gestos terá uma qualidade inferior ou não funcionará se a imagem fornecida inicialmente for de baixa qualidade.

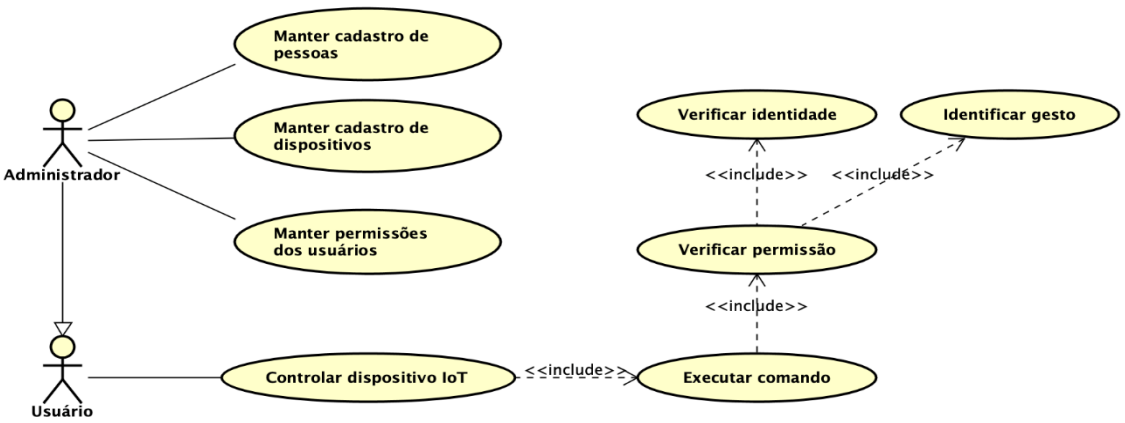
4.1 Casos de uso

Nesta subseção são demonstrados os casos de uso e possibilidades para os diferentes atores do sistema.

Para o administrador do sistema dentro de um contexto controlado, como em uma residência, há a necessidade de cadastrar as pessoas que poderão controlar o ambiente e também especificar quais dispositivos e ações cada pessoa terá acesso para comandar o ambiente, conforme apresentado na Figura 20.

Depois de realizado o cadastro das pessoas, é executado o treinamento com modelo baseado em Deep Learning para extração da biometria facial das múltiplas fotos de cada usuário. Assim quando algum usuário realizar gestos para controlar algum dispositivo, é verificado a permissão, que ocorre através da biometria facial e consequentemente são identificadas as ações permitidas, como apresentado no caso de uso da Figura 20. Assim que as devidas identificações são confirmadas, é executado a comunicação com o dispositivo IoT, através do módulo Execução.

Figura 20 - Casos de uso do sistema.



Fonte: Elaborado pelo autor (2019)

4.2 Requisitos

Os requisitos do sistema foram definidos com base na especificação do protótipo da solução proposta, assim como também elencados por meio da pesquisa referencial e estudo dos trabalhos relacionados. Os requisitos funcionais são apresentados pelo Quadro 1, e os requisitos não funcionais são apresentados pelo Quadro 2. Os requisitos definidos a seguir definem critérios para o correto desenvolvimento e validação do protótipo proposto.

Quadro 1 - Requisitos funcionais

RF01	
Nome	Manter cadastro de pessoas
Descrição	
O módulo controle deve possibilitar o cadastro de pessoas e principalmente a vinculação de múltiplas fotos de uma pessoa, tornando a aprendizagem mais assertiva o possível, ao aprender diferentes variações da face.	
RF02	
Nome	Manter cadastro de dispositivos
Descrição	
O módulo deve ser capaz de cadastrar diferentes dispositivos que possam ser controlados, pelo menos por acionamento e desligamento.	
RF03	
Nome	Manter cadastro de permissões
Descrição	
O sistema deve permitir controlar as permissões de maneira granular, sendo possível definir as ações cada pessoa de maneira independente.	
RF04	
Nome	Reconhecimento de gestos estáticos das mãos

Descrição	
O sistema deve ser capaz de reconhecer os gestos estáticos das mãos para que seja possível suportar diferentes comandos. Os gestos suportados precisam ser previamente treinados.	
RF05	
Nome	Identificação da biometria facial dos usuários
Descrição	
Para possibilitar a identificação das pessoas, o sistema precisa realizar a aprendizagem da biometria facial das pessoas sob modelo supervisionado com Deep Learning, para que posteriormente o sistema possa inferir a identidade do usuário que realiza a interação.	

Fonte: Elaborado pelo autor (2019)

Quadro 2 - Requisitos não funcionais

RNF01	
Nome	Linguagem de programação principal
Descrição	
O sistema precisa ser desenvolvido com Python 3 para melhor compatibilidade com o *framework* OpenCV.	
RNF02	
Nome	Módulo de Execução configurado para utilização de hardware modular
Descrição	
O módulo de execução precisa ser desenvolvido utilizando um hardware modular, por ser de baixo custo e possuir recursos moderados de hardware e conectividade.	
RNF03	
Nome	*Webservice* REST
Descrição	

O serviço REST disponibilizado no módulo de Execução deve ser desenvolvido em NodeJS, devido a facilidade de instalação, baixo consumo de recursos de hardware e de fácil configuração do servidor para lidar comunicação em HTTP/HTTPS.	
RNF04	
Nome	Servidor local
Descrição	
Para resolver problemas de processamento e *delay*, o sistema precisa ser disponibilizado localmente, a fim de buscar uma menor latência no processamento e execução dos comandos. Como *hardware* para disponibilizar o sistema, especifica-se o uso de hardwares modulares como NVIDIA Jetson focado para soluções de inteligência artificial, com aceleração em GPU e com grande capacidade para processamento paralelo (NVIDIA, 2019).	
RNF05	
Nome	Controle de segurança
Descrição	
Durante a interação o sistema precisa se assegurar de que nenhum comando seja executado por alguma pessoa não autorizada, assim realizando a identificação da pessoa por biometria facial.	
RNF06	
Nome	Múltiplos dispositivos IoT conectados
Descrição	
O módulo de Execução deve ser capaz de se conectar com múltiplos dispositivos IoT ao mesmo tempo.	
RNF07	
Nome	Aceitar diferentes fontes de entrada de imagem
Descrição	
A aplicação deve suportar a captura de diferentes câmeras para monitorar o ambiente na espera de comandos. Com resolução mínima aceitável é de 600x600px.	
RFN08	

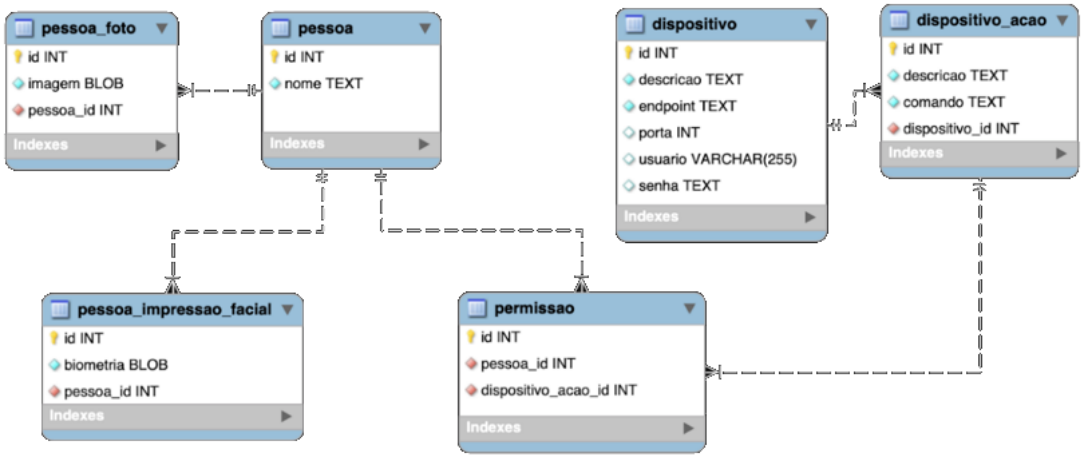
Aprimoramento das imagens	
Descrição	
O módulo de Controle precisa realizar a padronização das imagens fornecidas, garantindo pelo menos, imagens de mesma resolução como fonte de entrada para o algoritmo de visão computacional, a fim de se obter melhores resultados.	
RFN09	
Nome	Distância mínima e máxima para captura dos *frames*
Descrição	
Cada fonte de entrada precisa estar posicionada a uma distância mínima e máxima de 1 e 2.5 metros respectivamente, em relação ao sujeito que deseja realizar a interação.	
RFN10	
Nome	Gestos precisam ser confortáveis
Descrição	
A escolha dos gestos deve levar em consideração os critérios de usabilidade, não utilizando gestos que realizam rotação do pulso, preferencialmente com dedos levemente flexionados e próximos uns aos outros, evitando ao máximo a realização de gestos que possam causar desconforto.	

Fonte: Elaborado pelo autor (2019)

4.3 Diagrama entidade-relacionamento

O modelo de entidade-relacionamento do projeto segue uma estrutura relacional para armazenar e estruturar os dados da aplicação, conforme Figura 21, a estrutura contempla o armazenamento dos dados pessoais, bem como as múltiplas fotos de cada usuários e a aprendizagem da biometria facial de cada pessoa. Também para o funcionamento da aplicação se faz necessário registrar as permissões de cada usuário com os dispositivos que serão controlados.

Figura 21 - Modelo entidade-relacionamento do módulo de cadastros



Fonte: Elaborado pelo autor (2019)

O Quadro 3 detalha os objetivos de cada tabela apresentada na Figura 21.

Quadro 3 - Detalhamento das tabelas do banco de dados do módulo de cadastros.

Nome da tabela	Descrição
pessoa	Registra dados básicos de todas as pessoas que possuem acesso ao sistema.
pessoa_foto	Armazena todas as fotos de uma pessoa, para que seja possível realizar o treinamento com base na biometria facial, cada foto de um usuário representa um registro nesta tabela, sendo necessárias múltiplas fotos de uma mesma pessoa.
dispositivo	Representa o dispositivo IoT conectado na rede que será controlado.
dispositivo_acao	Representa as funcionalidades disponíveis do dispositivo IoT.
permissao	Armazena a permissão de uma pessoa em realizar uma ação em um determinado dispositivo.
pessoa_impressao_facial	Armazena a biometria facial aprendida pelo modelo baseado em Deep Learning.

Fonte: Elaborado pelo autor (2019)

4.4 Módulo Cadastro

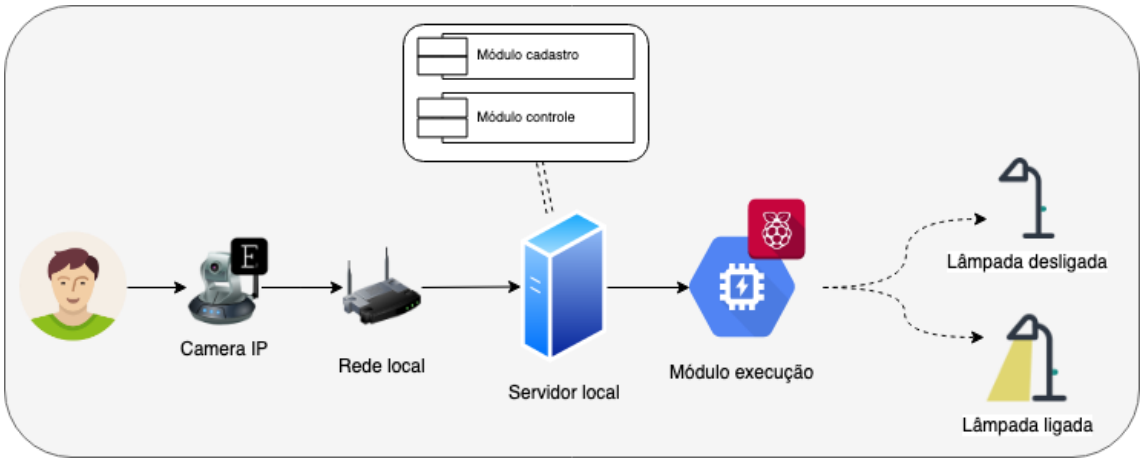
É responsabilidade deste módulo manter a parte administrativa, como os cadastros das pessoas, dispositivos e permissões, ou seja, todos as informações base necessárias para o correto funcionamento do módulo Controle. Neste módulo é realizado também o processo de limpeza e aprimoramento das imagens, a fim de garantir uma maior qualidade no aprendizado da biometria facial dos usuários. O treinamento e a correta disponibilização das informações são de responsabilidade deste módulo, para que os demais possam consumir os dados se assim for necessário.

Após a realização do treinamento com modelo baseado em Deep Learning, o modelo de reconhecimento da biometria facial das pessoas cadastradas é armazenado no banco de dados, para que quando o sistema iniciar, o módulo de Controle possa carregar os dados em memória, como meio de evitar etapas adicionais ao processo que possam aumentar o tempo de resposta, desde o processamento do algoritmo de visão computacional até o momento de acionar o dispositivo conectado.

4.5 Módulo Controle

Este módulo é encarregado de buscar e utilizar as informações básicas geradas no módulo Cadastro e realizar o processamento dos *frames* capturados pelas câmeras. Ao iniciar, este módulo carrega o modelo de reconhecimento facial das pessoas autorizadas, e assim que algum gesto for realizado, é identificado se a pessoa em questão possui permissão para executar a ação e caso positivo é realizado a comunicação com o dispositivo IoT cadastrado. O exemplo de interação com algum dispositivo IoT, pode ser visto na Figura 22, na qual dependendo do gesto utilizado uma pessoa autorizada pode ligar ou desligar uma lâmpada.

Figura 22 - Fluxo de interação para ligar ou desligar uma lâmpada da IoT



Fonte: Elaborado pelo autor (2019)

Assim após o processamento do algoritmo de visão computacional pelo módulo de Controle, é realizado a comunicação com o módulo de Execução que realiza a comunicação com o dispositivo IoT correspondente ao gesto. É importante ressaltar que dependendo dos dispositivos que serão conectados, pode-se utilizar diferentes módulos de Execução, pois o processamento principal estará centralizado no servidor local. Assim a simplicidade do módulo de Execução por apenas realizar a comunicação garante a modularidade e facilidade em adicionar ou substituir o módulo se assim for necessário.

4.6 Módulo Execução

O módulo de Execução é a última camada antes do acionamento direto dos dispositivos IoT que estão conectados. Este módulo disponibiliza por meio de NodeJS um *webservice* REST a ser consumido, sempre ativo e aguardando por comandos provenientes do módulo de Controle. Este módulo representa um *gateway* de comunicação com os dispositivos IoT, ele precisa ser simples para facilitar que seja substituído em caso de danos ou para que outro módulo de Execução seja adicionado para suprir a interface de comunicação de algum dispositivo IoT em específico, assim garantindo a modularidade e a conectividade necessária.

5 DESENVOLVIMENTO

Neste capítulo são apresentados detalhes do desenvolvimento do protótipo, assim como detalhes sobre as bibliotecas e tecnologias utilizadas para implementação do reconhecimento de gestos estáticos, biometria facial e controle de hardware modular.

5.1 Visão geral

Para validar o protótipo da solução proposta, foi desenvolvido um protótipo, onde que todos os módulos do projeto especificado foram desenvolvidos e disponibilizados em um notebook, não ocorrendo a captura de *frames* através da rede, mas a captura direta dos *frames* com a câmera integrada, o notebook possui 4GB de RAM e processador i5 com dois núcleos independentes.

O protótipo é composto basicamente por três módulos, conforme especificado anteriormente. Os módulos de Cadastro e Controle foram desenvolvidos juntamente como um módulo integrado. Para o módulo de Execução foi realizado a comunicação com o hardware modular Arduino Uno por meio da USB.

Ao iniciar, o módulo de Controle carrega o algoritmo para reconhecimento facial e de gestos estáticos, juntamente com o modelo já treinado para reconhecimento dos usuários e dos gestos estáticos. O algoritmo então realiza a captura de cada *frame* da câmera, redimensiona para o tamanho de 600x600 *pixels*. Este tamanho foi determinado por considerar-se suficiente

para a aplicação pretendida e porque resoluções superiores impactam diretamente no desempenho da aplicação.

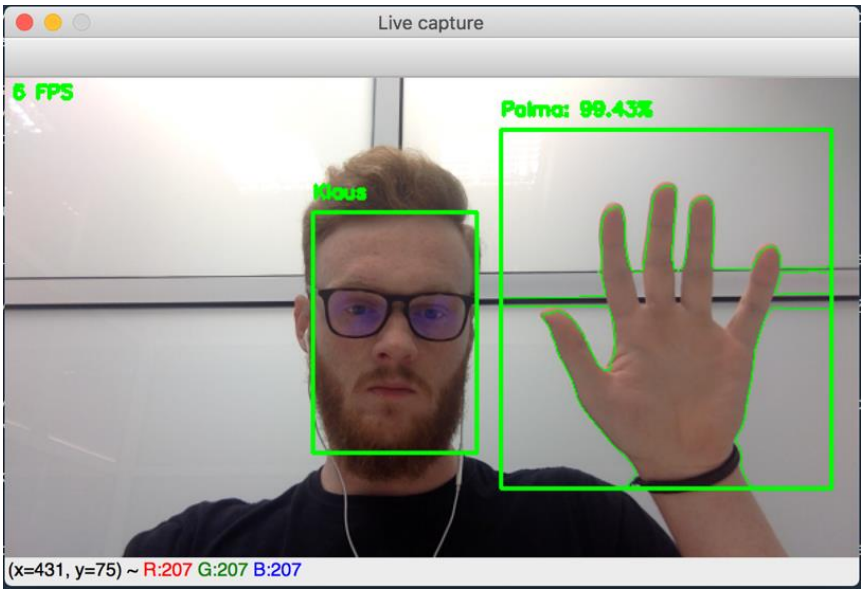
O módulo de Controle realiza a captura e processamento dos *frames*, que consiste na identificação facial e o reconhecimento do gesto estático. Após a identificação correta de um usuário cadastrado é realizado a identificação do gesto e a comunicação com o módulo de Execução por meio de chamadas REST.

O módulo de Execução disponibiliza um *webservice* REST em NodeJS, como um *gateway* de comunicação com diferentes dispositivos, sendo realizado a comunicação com o hardware modular Arduino Uno para o acionamento das saídas digitais conforme configuração enviada pelo módulo de Controle.

5.2 Interface do protótipo

A interface visual do protótipo foi desenvolvida para o propósito de verificação dos resultados e desempenho geral da solução prototipada. Na interface do protótipo é possível visualizar a detecção das principais áreas de interesse para o processamento do algoritmo de visão computacional, como a área onde que a face e a mão se encontram. Cada área de interesse para processamento do algoritmo é conhecida como Region of Interest (ROI). Para a ROI facial detectada na interface também é exibido o nome do usuário se corretamente identificado como válido, em caso de não reconhecimento a face é definida como “Desconhecida”. Além da identificação da face, também é possível visualizar a ROI com a detecção do gesto quando o mesmo for identificado como conhecido. A interface do protótipo, assim como os reconhecimentos realizados e a medição do desempenho de 5fps pode ser visualizada na Figura 23.

Figura 23 – Interface com identificação facial, gesto e velocidade do processamento



Fonte: Elaborado pelo autor (2019)

5.3 Cadastro de usuários

Para realizar o cadastro dos usuários no protótipo, responsabilidade do módulo Cadastro, foi desenvolvido um algoritmo, juntamente com a interface visual do protótipo, onde que é possível iniciar a inserção de um novo usuário mediante o pressionamento da tecla “N”, assim o algoritmo irá esperar até que o nome para identificação do novo usuário seja informado, ao confirmar o nome, o sistema iniciará a captura da face da pessoa, sendo que para cada cadastro são capturadas múltiplas imagens da mesma pessoa. São armazenados em uma pasta específica somente a ROI, com a face centralizada de cada imagem do novo usuário. O treinamento das faces cadastradas ocorre mediante a execução manual de um *script* em Python que inicia o treinamento, realizando a extração das características faciais como fonte de entrada para o algoritmo de treinamento, como o Fisherfaces do OpenCV ou o modelo Deep Learning com o *framework* Dlib. Para que os novos usuários possam ser identificados em tempo real é necessário que o sistema seja inicializado novamente após a realização do treinamento.

5.4 Detecção facial

A detecção facial foi implementada fazendo uso do algoritmo baseado em Histogram of Oriented Gradients (HOG) do *framework* Dlib. O processo para a detecção facial inicia com a conversão da imagem para preto e branco (não é necessário o maior detalhamento de cores para este algoritmo) e na imagem convertida é aplicado o algoritmo de detecção, sendo obtidos como resultado as ROI de todas as pessoas presentes no *frame* e para cada ROI é realizado o reconhecimento facial que é detalhado na seção seguinte.

5.5 Reconhecimento facial

Durante o desenvolvimento da funcionalidade de identificação facial do protótipo foram realizados testes com dois algoritmos distintos, com o objetivo de obter o melhor resultado e a menor taxa de falso negativo e falso positivo. O primeiro algoritmo utilizado foi o Fisherfaces disponibilizado pelo OpenCV, porém, o melhor resultado foi obtido utilizando o modelo de Deep Learning para extração das características faciais disponível pelo *framework* Dlib.

Dois usuários foram utilizados para a validação do protótipo, sendo que quantidades diferentes de fotos foram utilizadas para o treinamento de cada usuário, para o primeiro foram utilizadas 129 fotos e 58 para o segundo.

A seguir será detalhado o processo para treinamento realizado com os dois algoritmos, os resultados obtidos quanto ao reconhecimento facial podem ser visualizados no próximo capítulo.

5.5.1 Reconhecimento facial com OpenCV

A extração das características faciais e o reconhecimento facial com OpenCV foi implementado utilizando o algoritmo Fisherfaces. O reconhecedor do OpenCV ao realizar o reconhecimento facial retorna dois valores, sendo o primeiro o *id* da classe que representa a

identidade do usuário e o segundo representa taxa de semelhança do reconhecimento, denominada de *confidence.* Juntamente com o valor de semelhança se faz necessário o uso de um valor denominado *threshold* que define um valor máximo para a taxa de semelhança, assim assumindo como verdadeiro o reconhecimento onde que a taxa de semelhança for menor que o valor *threshold*.

O *threshold* necessário pode variar dependendo da resolução das imagens utilizadas para o reconhecimento, o limite para este valor utilizado foi de 4000 com base em faces de tamanho 150x180 *pixels*.

A extração das características faciais e o aprendizado com o reconhecedor foi realizado da seguinte maneira para cada imagem cadastrada dos usuários:

1. Conversão da imagem para o esquema de cores preto e branco;
2. Detecção da ROI - posição da face;
3. Normalização da ROI;
4. Face normalizada é inserida e o processo de aprendizagem é realizado com o reconhecedor Fisherfaces do OpenCV;
5. Os dados extraídos do reconhecimento facial são salvos separadamente para a posterior utilização do modelo ensinado.

Após a detecção da ROI facial é iniciado o processo de normalização que visa padronizar a imagem, garantindo um tamanho único e realizando a equalização de cores em todas as imagens utilizadas. A ROI tem seu tamanho redimensionado conforme a documentação do OpenCV, porém a mesma não é clara sobre o algoritmo utilizado para a redução do tamanho realizada pelo parâmetro INTER_AREA. Já o aumento de imagens que não atendem o tamanho esperado ocorre com uma interpolação bicúbica que é realizada pelo parâmetro INTER_CUBIC (OPENCV, 2019).

Na etapa final da normalização é realizada a equalização do histograma de cores, que visa aumentar o contraste, realizando o equilíbrio entre a faixa de intensidade das cores. O melhor contrate da imagem visa tornar as características faciais mais evidentes para que o

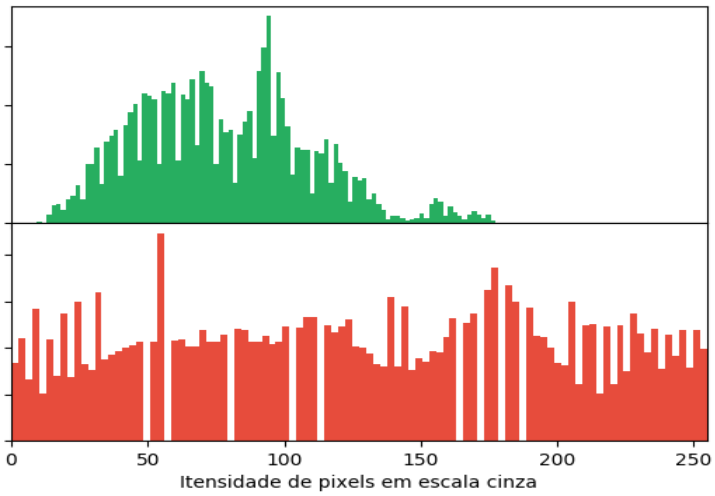
modelo de extração das características obtenha melhores resultados. Conforme a Figura 24, é possível visualizar a diferença antes e após a equalização de cores.

Figura 24 - Resultado da equalização de cores e aumento do contraste

Fonte: Elaborado pelo autor (2019)

Conforme a Figura 25 é possível verificar a diferença entre os níveis de cores da imagem, o gráfico superior representa a imagem original onde apenas a conversão para preto e branco foi realizada, já no gráfico inferior é demonstrado o resultado juntamente com a equalização de cores.

Figura 25 - Gráfico comparando o esquema de cores após equalização



Fonte: Elaborado pelo autor (2019)

Uma etapa importante no processo de normalização da face que visa aumentar a qualidade é a rotação, pois é comum que mediante a captura em tempo real os usuários mudem sua posição ou inclinação de suas faces para posições diferentes das imagens que foram utilizadas durante o treinamento.

Pelo fato do algoritmo de detecção facial capturar exatamente a ROI com a face centralizada, é necessário que antes de iniciar esta etapa a ROI seja aumentada em alguns *pixels* em todos os sentidos, afim de que seja possível realizar a etapa de rotação, caso contrário a face seria cortada e detalhes sobre a face seriam perdidos durante a normalização.

O algoritmo para rotação das faces foi desenvolvido utilizando um classificador previamente criado pelo OpenCV para a detecção da posição dos olhos. Assim frente a diferença de altura que existe entre os olhos é possível criar uma matriz de transformação que realize a rotação em sentido contrário.

Porém para a utilização deste algoritmo no processo de normalização é necessário que pelo menos duas detecções a mais ocorram, uma para a identificação dos olhos e outra final para realizar a detecção facial após a rotação da face afim de que apenas a ROI facial rotacionada seja utilizada. Desta maneira houve um considerável aumento de tempo no processo de normalização da face, assim não sendo viável sua utilização em tempo real para o processamento de todos os *frames* e consequentemente foi desativado do processo de normalização.

5.5.2 Reconhecimento facial com Dlib

A utilização deste algoritmo baseado em Deep Learning ocorreu de maneira mais simples, se comparado com a identificação realizada com o OpenCV, pois os resultados até então já eram superiores e adequados para a homologação do protótipo, sendo assim a etapa de normalização não foi realizada. O preparo de cada imagem e treinamento ocorreu conforme as seguintes etapas:

1. Conversão da imagem para o esquema de cores preto e branco;
2. Detecção da ROI - posição da face;
3. Face detectada em esquema de cores RGB é inserida e o processo de aprendizagem é realizado com o reconhecedor baseado em Deep Learning;
4. Os dados extraídos do reconhecimento facial são salvos separadamente para a posterior utilização do modelo ensinado.

O algoritmo disponibilizado pelo *framework* Dlib é baseado em um modelo de Deep Learning que foi treinado em uma base de imagens em torno de 3 milhões de faces para ser capaz de extrair e definir 128 medidas denominadas de *embeding* que quantificam as características faciais de uma pessoa. O treinamento deste algoritmo ocorre por um processo conhecido como Triplet Training Step, onde que por meio de três imagens únicas, sendo duas da mesma pessoa e outra de uma pessoa diferente, é criado o vetor dimensional de 128 pontos. O peso entre a distância das características faciais das duas fotos da mesma pessoa é aproximado e terceira face diferente serve para distanciar os valores do que seria o conjunto de dados de uma pessoa da outra (GEITGEY, 2016).

Este algoritmo, diferente do utilizado no Fisherfaces do OpenCV, utiliza o esquema de cores em formato RGB, não sendo necessário a conversão para preto e branco, sequer o ajuste de contrataste. A ROI com a face de cada usuário não passou pelo processo de normalização que foi realizado no treinamento com o algoritmo Fisherfaces. O *embeding* facial da face é extraído e comparado com o modelo previamente treinado no módulo Cadastro, o resultado do reconhecimento facial deste algoritmo, diferente dos reconhecedores do OpenCV, retorna uma lista de candidatos, sendo necessário filtrar pelo candidato que recebeu mais votos para se obter a identidade da pessoa. Este algoritmo também se diferencia por não necessitar um limite

threshold, a recomendação para o reconhecimento leva em consideração que se a distância euclidiana do vetor da face da pessoa for menor que 0.6 em comparação com outro vetor então os vetores são da mesma pessoa. No protótipo foi utilizado como base 0.5 para a distância euclidiana ao realizar a comparação e identificação dos usuários, este parâmetro é definido ao solicitar o reconhecimento da face, o valor padrão configurado é de 0.6 para a distância euclidiana máxima, assim utilizando valores inferiores a rigorosidade do reconhecimento é aumentada.

O framework Dlib possui suporte para processamento em GPU e para sua ativação é necessário que o *framework* seja compilado, porém não foi possível ativar o processamento por GPU pelo fato de que o hardware utilizado para o desenvolvimento do protótipo não possui os requisitos mínimos, como placa gráfica NVIDIA com suporte a API CUDA. Assim toda a utilização das funções do *framework* Dlib ocorreram mediante o processamento em CPU.

5.6 Reconhecimento de gestos estáticos

Para o reconhecimento de gestos estáticos foi feito o uso de um projeto público, chamado “Hand Gesture Recognition Using Background Ellimination and Convolution Neural Network” (HGRUBECNN), que é baseado em Deep Convolution Neural Network, uma rede neural profunda que realiza o reconhecimento de gestos estáticos das mãos (SAHA, 2019). O projeto em questão possui uma rede neural configurada, não observando-se necessidade de modificação de sua configuração padrão. No entanto, devido a diferença entre as versões dos pacotes em Python do protótipo desenvolvido e do projeto HGRUBECNN, foi necessário que o treinamento fosse executado novamente. A execução do treino com a nova versão dos pacotes se fez possível pelo fato das imagens utilizadas inicialmente para treinamento estarem publicamente disponíveis.

O projeto HGRUBECNN inclui imagens para treinamento e modelo para realizar o reconhecimento de três gestos estáticos distintos, conforme Figura 26, sendo eles: punho, palma da mão e *swing*, respectivamente. Para que seja possível reconhecer mais gestos é necessário que sejam preparadas imagens dos gestos desejados e executado novamente o treinamento, assim sendo possível utilizar mais gestos para a interação.

Figura 26 - Gestos possíveis para reconhecimento

Fonte: Do autor, adaptado de SAHA (2019)

Uma etapa de extrema importância para o reconhecimento de gestos estáticos é a segmentação da ROI, visando que o fundo seja abstraído e apenas a região da mão fique visível, assim sendo possível remover detalhes da imagem que poderiam atrapalhar o treinamento e o reconhecimento (MATHWORKS, 2019). Porém, o projeto HGRUBECNN utiliza segmentação baseada em *binary thresholding* que sofre grande influência das condições de luminosidade presentes no ambiente. Para casos de diferentes luminosidades seria mais adequado a utilização de *adaptive thresholding*, a diferença entre ambas é demonstrada na Figura 27. O algoritmo para realização da segmentação com *adaptive thresholding* está disponível pela API do OpenCV, porém não foi possível adaptar esta etapa do reconhecimento implementado no projeto HGRUBECNN pois as imagens para treinamento foram convertidas utilizando *binary thresholding* e as imagens originais não estão disponíveis para que novos testes com um *thresholding* diferente seja facilmente utilizado (OPENCV, 2019).

Figura 27 - Resultado com *adaptative* e *binary thresholding* respectivamente

Fonte: Elaborado pelo autor (2019)

O projeto HGRUBECNN não realiza a detecção da mão, mas a captura de uma região fixa nos *frames*. Para avançar neste sentido, foi adicionado ao protótipo, o uso de dois classificadores Haar, publicamente disponíveis pelo OpenCV. Eles servem para a detecção de quando a mão está aberta e fechada, assim o processo de detecção da posição da mão no pior cenário pode necessitar que os dois classificadores sejam executados, pois se não é possível detectar a mão com o primeiro classificador, é realizado novamente o processo com o segundo

e se não forem encontrados resultados assume-se de que não existe uma mão na região do *frame* em processamento.

Como etapa final de normalização da ROI, é realizado um aumento de cerca de 20 *pixels* em torno da região detectada, pois o tamanho da mão ao realizar o gesto pode ser maior do que a ROI detectada pelos classificadores em cascata, assim evita-se de que a imagem fique cortada e não seja possível de identificar o gesto realizado. Por final a ROI é redimensionada para 89x100 *pixels*, formato esperado para a realização da detecção do gesto estático pelo modelo de reconhecimento baseado em CNN. Como resultado do reconhecimento há dois valores, o *id* da classe referente ao gesto e em segundo lugar o valor que define a taxa de assertividade denominada de *confidence*, assim assume-se como verdadeiro os resultados acima de 0.99, sendo a assertividade máxima possível definida pelo valor 1.

5.7 Acionamento de dispositivos

O acionamento de dispositivos à cargo do módulo de Execução foi implementado através de um serviço REST desenvolvido em NodeJS, gerenciando o recebimento das chamadas HTTP e como resultado final é realizado o acionamento ou desligando de algum dispositivo conectado às saídas digitais ou analógicas de um Arduino Uno.

Para a validação do protótipo foi feito o uso do hardware modular Arduino Uno conectada por meio da USB, porém outras placas poderiam ser utilizadas. O interessante neste módulo é a utilização do *framework* Johnny-Five que fornece uma camada de abstração para desenvolvimento de robótica e também plataformas IoT, assim de maneira simples, é possível que diferentes hardwares modulares possam ser utilizados, seguindo uma API única de comunicação.

Para o controle de dispositivos IoT diferentes é necessário que seja desenvolvido uma camada de comunicação com o dispositivo desejado, pois diferentes dispositivos podem seguir padrões diferentes, assim para o protótipo apenas uma camada API foi desenvolvida para intermediar comandos com as saídas digitais do Arduino Uno com o uso do *framework* Johnny-Five.

Para a utilização do Arduino Uno com o *framework* Johnny-Five é necessário que o *firmdata* padrão do Arduino seja carregado para o sistema embarcado. Após esta definição não é necessário que sejam definidas quaisquer configurações adicionais, o *framework* Johnny-Five já é capaz de reconhecer e controlar o hardware modular.

Para validação do protótipo, foi estipulada a comunicação com dispositivos onde fosse possível o controle por meio de dois comandos, como acionamento e desligamento. Para tanto, foi desenvolvido uma rota REST para comunicação que aguarda dois parâmetros, para definição de qual dispositivo será controlado foi estipulado o parâmetro *device*. E por final o parâmetro *activate* onde que dois valores são esperados, 1 para a situação de acionamento e 0 para o desligamento.

Para a validação deste protótipo e gerenciamento das ações provenientes dos gestos estáticos reconhecidos, foram utilizados apenas dois gestos. Para o acionamento foi utilizado o gesto referente ao estado de mão aberta e para o desligamento o gesto onde que a mão se encontra fechada, não sendo necessário a utilização do terceiro gesto *swing*. Como teste deste sistema foi definido que as interações dos gestos controlariam um LED, conforme Figura 28, sendo evidente o acionamento e desligamento por meio de seu sinal luminoso.

Figura 28 – Acionamento LED

Fonte: Elaborado pelo autor (2019)

A Figura 29, apresenta o código-fonte responsável pelo gerenciamento das ações de acionamento e desligamento realizadas no dispositivo.

Figura 29 – Trecho de código responsável pelo acionamento e desligamento de dispositivos

```
board.on('ready', function() {
    app.get('/handle/:device/:activate', (req, res) => {
        const { device, activate } = req.params;

        const deviceInstance = getDeviceInstance(device);

        if(activate == 1) {
            deviceInstance.on();
            res.json({activate: true});
            console.log(`Activating device ${device}...`);
        } else {
            deviceInstance.off();
            res.json({activate: false});
            console.log(`Deactivating device ${device}...`);
        }
    });

    serverInstance = app.listen(port, () => console.log(`Listening on port ${port}!`));
});
```

Fonte: Elaborado pelo autor (2019)

6 AVALIAÇÃO DO PROTÓTIPO E RESULTADOS

Neste capítulo serão apresentados os resultados obtidos a partir do protótipo desenvolvido. Para validação dos algoritmos de reconhecimento foram coletadas e analisadas novas imagens, diferentes das utilizadas para o treinamento nos algoritmos, a fim de medir suas assertividades. Por fim, foi realizada a medição do tempo médio para processamento individual dos algoritmos e o tempo total médio para processamento de cada *frame*. Com exceção da Amostra 3 do reconhecimento facial, todos os testes a seguir foram realizados em ambiente controlado de luminosidade e com pelo menos dois usuários.

6.1 Posicionamento da câmera

Para a captura correta das imagens é necessário que a câmera seja posicionada corretamente no local de interesse, pois a interação em situações de demasiado longe, acima de 2.5 metros da câmera, ou demasiado perto, menos de 1 metro da câmera, não é possível obter bons resultados. Para a situação de maior distância, o reconhecimento de gestos estáticos perde seu percentual de *confidence*, não atingindo o mínimo esperado para aceitar o gesto. E para a situação muito próxima à câmera não se recomenda, pois, os classificadores para detecção da mão não reconhecem corretamente a mesma, uma vez que muito próxima, a mão tem um tamanho maior do que o esperado para detecção dos classificadores. Alteração para detecção da mão é possível, porém impactaria também desempenho final.

6.2 Reconhecimento facial

Nesta seção serão analisados os resultados dos testes realizados com os algoritmos de biometria facial utilizados no protótipo, foram coletadas e testadas 3 amostras de imagens, cada imagem das amostras foi submetida aos dois modelos de reconhecimento facial treinados durante o desenvolvimento do protótipo.

6.2.1 Amostra 1

Para esta amostra foram coletadas 100 imagens faciais de um usuário, sendo estas imagens diferentes das 58 imagens utilizadas para seu treinamento. Os resultados estão dispostos na Tabela 1.

Tabela 1 – Resultado da Amostra 1 – reconhecimento do primeiro usuário

	Deep Learning com Dlib	**Fisherfaces com OpenCV**
Verdadeiro positivo	100	75
Falso negativo	0	25

Fonte: Elaborado pelo autor (2019)

O teste para esta amostra aponta melhores resultados para o modelo baseado em Deep Learning, onde 100% da amostragem foi identificada corretamente, para o algoritmo Fisherfaces houve uma taxa de 25% de falso negativo, causando a não identificação do usuário.

6.2.2 Amostra 2

Para esta amostra também foram coletadas 100 imagens faciais do segundo usuário, sendo estas imagens diferentes das 129 imagens utilizadas para seu treinamento. Os resultados estão dispostos na Tabela 2.

Tabela 2 – Resultado da Amostra 2 – reconhecimento do segundo usuário

	Deep Learning com Dlib	Fisherfaces com OpenCV
Verdadeiro positivo	100	16
Falso negativo	0	84

Fonte: Elaborado pelo autor (2019)

Os testes realizados sobre esta amostra apontam novamente melhores resultados para o modelo baseado em Deep Learning, onde 100% da amostragem foi identificada corretamente, já para o algoritmo Fisherfaces houve um alto valor para falso negativo. É interessante ressaltar que apesar do maior número de imagens utilizadas para treinamento se comparado com o usuário da Amostra 1 os resultados não foram superiores.

6.2.3 Amostra 3

Para esta amostra foram utilizadas 272 imagens faciais de pessoas desconhecidas e com diferentes condições de luminosidade, tendo como objetivo a identificação dos melhores resultados para verdadeiro negativo e falso positivo, assim sendo possível verificar qual algoritmo obteve melhores resultados na identificação de pessoas que não pertencem ao conjunto de usuários cadastrados e também se o algoritmo reconhece impostores como usuários verdadeiros do sistema. Os resultados estão dispostos na Tabela 3.

Tabela 3 – Resultados da Amostra 3 – identificação de impostores

	Deep Learning com Dlib	Fisherfaces com OpenCV
Verdadeiro negativo	269	225
Falso positivo	3	47

Fonte: Elaborado pelo autor (2019)

Com base nos resultados é possível identificar que o algoritmo baseado em Deep Learning obteve excelentes resultados na identificação correta de usuários impostores do sistema, com apenas uma pequena taxa de 1.10% para falso positivo. Enquanto o algoritmo Fisherfaces obteve um alto valor para o reconhecimento de impostores como usuários válidos do sistema, com uma taxa de 17.28%.

6.3 Reconhecimento de gestos estáticos

Nesta seção serão analisados os resultados dos testes com o algoritmo de reconhecimento de gestos estáticos utilizado no protótipo, foram coletadas e testadas 3 amostras de imagens diferentes das utilizadas para o treinamento.

6.3.1 Amostra 1 – palma da mão

Para esta amostra foram coletadas 100 imagens onde que a palma da mão era o gesto realizado, os resultados estão dispostos na Tabela 4.

Tabela 4 – Resultados da Amostra 1 – palma da mão

	Resultados
Verdadeiro positivo	100
Falso negativo	0

Fonte: Elaborado pelo autor (2019)

Conforme os testes, foi identificado alta taxa de reconhecimento para a palma da mão, porém a eficácia deste resultado será discutida melhor na Amostra 3.

6.3.2 Amostra 2 – punho da mão

Para esta amostra foram coletadas 100 imagens onde que o punho da mão era o gesto realizado, os resultados estão dispostos na Tabela 5.

Tabela 5 – Resultados da Amostra 2 – punho da mão

	Resultados
Verdadeiro positivo	76
Falso negativo	24

Fonte: Elaborado pelo autor (2019)

Diferente do reconhecimento da palma da mão, o modelo não foi capaz de reconhecer todas as imagens desta amostra como verdadeiros, apresentando uma taxa de 24% para falso negativo, não sendo capaz de reconhecer os gestos de punho como válidos do modelo.

6.3.3 Amostra 3 – gestos desconhecidos

Para esta amostra foram coletadas 300 imagens onde que o gesto da mão é desconhecido para o modelo, assim sendo possível verificar se o modelo reconhece corretamente gestos inválidos como tais e não pertencentes a nenhum gesto conhecido do modelo. Os resultados estão dispostos na Tabela 6.

Tabela 6 – Resultados da Amostra 3 – gestos desconhecidos

	Resultados
Verdadeiro negativo	232
Falso positivo	68

Fonte: Elaborado pelo autor (2019)

Conforme os resultados obtidos, fica evidente o alto número para falso positivo, ou seja, o modelo ainda não é capaz de distinguir de maneira adequada gestos desconhecidos dos gestos treinados e consequentemente há uma incorreta classificação de um gesto inválido como algum válido do modelo, apresentando uma taxa de 22.67% para falso positivo. As imagens utilizadas para o teste possuem em sua maioria pelo menos algum dos dedos em evidência, assim o modelo acabou reconhecendo gestos com mais de um dedo em evidência como a palma da mão. Assim o resultado da Amostra 1 se torna mais evidente, o modelo acaba sendo tendencioso a classificar algum gesto com mais dedos como o sendo gesto da palma da mão onde todos os 5 dedos estão visíveis.

6.4 Desempenho geral do protótipo

O desempenho geral do protótipo foi calculado com base em uma média de 200 execuções procedurais em cada etapa do algoritmo, sendo considerado apenas situações positivas, onde que tanto o reconhecimento de gesto estático quanto a identificação facial do usuário estavam corretos, os valores podem ser visualizados na Tabela 7.

Tabela 7 - Tempo para processamento de cada algoritmo do protótipo

Etapa	Tempo (milissegundos)
Reconhecimento facial	~162
Reconhecimento de gestos	~58
Execução de dispositivo	~11
Tempo total	~291

Fonte: Elaborado pelo autor (2019)

Quanto ao tempo de processamento, o protótipo apresentou bons resultados, sendo possível completar o processamento de pelo menos 3 *frames* por segundo. Tanto a identificação facial quanto o reconhecimento de gestos estáticos ocorrem a cada *frame* processado, isto é possível pelo fato de ser realizado o reconhecimento de gestos estáticos das mãos e não gestos

em movimento que demandam o processamento de múltiplos *frames* em sequência para identificação do gesto realizado. O consumo dos recursos de hardware para executar o protótipo foi de 100% de um processador e 10% do segundo, com uma utilização de 400 MB de memória RAM.

.

7 CONCLUSÕES

Com o objetivo de prototipar uma solução de interação com dispositivos em casas inteligentes, o presente trabalho apresentou o desenvolvimento de um protótipo que possibilita a interação com dispositivos conectados por meio do reconhecimento de gestos estáticos das mãos, garantindo segurança durante a interação por meio da identificação biométrica facial dos usuários.

Através do desenvolvimento do protótipo foi possível integrar diferentes algoritmos de diferentes tecnologias para o desenvolvimento de uma solução para interação com o ambiente, como em casas inteligentes. Através dos testes realizados foi possível constatar excelentes resultados do modelo baseado em Deep Learning para reconhecimento facial se comparado ao algoritmo de reconhecimento facial Fisherfaces. O protótipo desenvolvido apresentou bons resultados de desempenho para o processamento em tempo real de cada *frame*, sendo possível realizar o processamento completo de pelo menos 3 *frames* por segundo.

As principais dificuldades durante o desenvolvimento do protótipo estiveram relacionadas às estratégias para abordar e solucionar os problemas de visão computacional em geral, como segmentação, detecção correta das ROI e qualidade dos resultados obtidos. Além disto, foi necessário realizar diferentes testes com diferentes resoluções na busca de um bom desempenho sem perda na qualidade do reconhecimento facial, embora no final, os piores resultados estiveram relacionados ao reconhecimento de gestos estáticos, onde foi necessário a utilização de diferentes classificadores para detectar as mãos e a busca por modelos capazes de reconhecer corretamente os gestos realizados.

Embora a abordagem satisfaça os objetivos propostos pelo trabalho, existem melhorias que poderiam ser desenvolvidas e algumas estratégias repensadas para o desenvolvimento de uma solução com maior desempenho, usabilidade no cadastro de usuários e precisão no reconhecimento de gestos estáticos.

7.1 Trabalhos Futuros

Pelo fato de o trabalho proposto demandar o estudo e conhecimento de diferentes tecnologias, não foi possível desenvolver todos os módulos propostos com plenitude, sendo assim necessário adotar estratégias que levaram a uma abordagem mais simples para o desenvolvimento do protótipo em virtude do curto prazo para finalização do trabalho.

Por questões de segurança é necessário de que quando existam mais de uma pessoa em uma mesma ROI do *frame* processado o sistema deve ignorar os comandos, a fim de evitar que uma pessoa não autorizada realize comandos sem permissão quando um usuário do sistema está presente. Mediante a testes com o detector do OpenCV para detecção de pessoas não foi possível realizar este controle, pois classificador identifica pessoas de corpo inteiro e como os sujeitos da interação podem estar mais próximos à câmera não foi possível realizar este controle tão facilmente.

Por questões de usabilidade, se faz necessário o desenvolvimento de uma interface administrativa mais amigável para cadastramento de pessoas e controle de permissão dos dispositivos a serem consumidos.

A realização de testes com câmeras de profundidade, como Leap Motion ou Kinect podem resolver os problemas relacionados à captura, segmentação e identificação correta de toda a estrutura das mãos, assim facilitando desenvolvimento do protótipo proposto e possivelmente o reconhecimento de gestos estáticos.

Como o sistema propõe o reconhecimento de gestos estáticos como meio de interagir com o ambiente e dispositivos, se faz necessário de que mais gestos sejam reconhecidos, sendo necessário o treinamento e disponibilização de mais comandos para utilização. Uma alternativa para mitigar este problema, seria definir o ambiente ou contexto no qual a câmera está instalada juntamente com a ação, de modo que um gesto possua diferentes ações dependendo do contexto aonde ele foi identificado. Desta maneira, com a utilização de apenas um gesto de acionamento,

seria possível ligar uma *smart TV* quando identificado no contexto da sala, mas também possibilitando controlar o acionamento do portão na garagem. Para melhorar a qualidade do reconhecimento de gestos define-se como melhoria a troca de segmentação para o uso de *adaptive thresholding*, assim buscando melhores resultados para o modelo de reconhecimento.

REFERÊNCIAS

AJIT, Jaokar. **An Introduction to Deep Learning and it's role for IoT/ future cities**. Disponível em: <https://www.datasciencecentral.com/profiles/blogs/an-introduction-to-deep-learning-and-it-s-role-for-iot-future>. Acesso em: 17 de novembro, 2019.

AMAZON. **What is Amazon Go?**. Disponível em: <https://www.amazon.com/b?ie=UTF8&node=16008589011>. Acesso em: 23 de maio, 2019.

ARDUINO. **Getting Started with Arduino UNO**. Disponível em: <https://www.arduino.cc/en/Guide/ArduinoUno>. Acesso em: 11 de novembro, 2019.

BENYON, David. **Designing Interactive Systems:** A comprehensive guide to HCI, UX and interaction design. 3 ed. Person, 2019.

BOECHAT, Gláucya C. **Investigação de um modelo de arquitetura biométrica multimodal para identificação facial**. Universidade Federal de Pernambuco, 2008.

BROWNLEE, Jason. **A Gentle Introduction to Pooling Layers for Convolutional Neural Networks**. Disponível em: <https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/>. Acesso em: 17 de novembro, 2019.

BROWNLEE, Jason. **What is Deep Learning?**. Disponível em: <https://machinelearningmastery.com/what-is-deep-learning/>. Acesso em: 17 de novembro, 2019.

CÁRDENAS, Edwin J. E. **Desenvolvimento de uma Abordagem para o Reconhecimento de Gestos Manuais Dinâmicos e Estáticos**. Universidade Federal de Ouro Preto, 2015.

CISCO. **IP Video Surveillance Design Guide**. Disponível em: <https://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Video/IPVS/IPVS_DG/IPVS-DesignGuide/IPVSchap4.html>. Acesso em: 16 de novembro, 2019.

COSTA, Silvia M. F. **Classificação e verificação de impressões digitais.** 2001. 123 f. Dissertação (Mestrado em Engenharia Elétrica) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2001.

DALAL, Navneet; TRIGGS, Bill. **Histograms of Oriented Gradients for Human Detection**. Disponível em: <http://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf>. Acesso em: 9 de novembro, 2019.

DAWSON-HOWE, Kenneth. **A Practical Introduction to Computer Vision with OpenCV**. 1. ED. John Wiley & Sons, 2014.

DICKSON, Ben. **Machine learning will be key to securing IoT in smart homes**. Disponível em: <https://bdtechtalks.com/2016/07/07/machine-learning-will-be-key-to-securing-iot-in-smart-homes>. Acesso em: 15 de novembro, 2019.

DUBEY, Isha S.; VERMA, Jyotsna S.; MEHENDALE, Arundhati. **An Assistive System for Visually Impaired using Raspberry Pi**. Disponível em: <https://www.academia.edu/39269567/IJERT-An_Assistive_System_for_Visually_Impaired_using_Raspberry_Pi>. Acesso em: 16 de novembro, 2019.

GALIMBERTI, Luiz H. O. **Estudo Comparativo De Algoritmos De Biometria Facial Disponibilizados Pela Biblioteca OpenCV para Controle de Acesso**. Univates, 2018.

GARTNER. **Gartner Identifies Top 10 Strategic IoT Technologies and Trends**. Disponível em: <https://www.gartner.com/en/newsroom/press-releases/2018-11-07-gartner-identifies-top-10-strategic-iot-technologies-and-trends>. Acesso em: 15 de novembro, 2019.

GEITGEY, Adam. **Machine Learning is Fun! Part 4:** Modern Face Recognition with Deep Learning. Disponível em: <https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78>. Acesso em: 9 de novembro, 2019.

GIL, Antonio. C. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2006.

GOMES G. S, BERGAMO F. V. M. **Chegou a Era da Internet das Coisas?** Um Estudo sobre Adoção de Objetos Inteligentes no Contexto Brasileiro, 2018. Disponível em: <http://www.revistabrasileiramarketing.org/ojs-2.2.4/index.php/remark/article/view/3648#?>. Acesso em: 18 de maio, 2019.

GONÇALVES, Luís P. N. **Reconhecimento de Gestos para Interação com Robô Móvel em Ambiente Pediátrico**. FCT, 2017.

ITU-T Study Group. **New ITU standards define the Internet of Things and provide the blueprints for its development**. ITU, 2012.

JOHNNY-FIVE. **JavaScript Robotics & IoT Platform**. Disponível em: <http://johnny-five.io/>. Acesso em: 11 de novembro, 2019.

MAGALHÃES, Paulo S.; SANTOS, Henrique D. **Biometria e autenticação**. Actas da 4ª Conferência da Associação Portuguesa de Sistemas de Informação, 2003.

MATHWORKS. **Image Segmentation**. Disponível em: <https://www.mathworks.com/discovery/image-segmentation.html>. Acesso em: 15 de novembro, 2019.

MATHWORKS. **Image Thresholding**. Disponível em: <https://www.mathworks.com/discovery/image-thresholding.html>. Acesso em: 9 de novembro, 2019.

MITANGI, Patel; PANDYA, Sharnil; PATEL, Satvik. **Hand Gesture based Home Control Device using IoT**. Disponível em: <https://www.ijarcs.info/index.php/Ijarcs/article/view/3659>. Acesso em: 16 de novembro, 2019.

MORAES, Jairo. **Controle de acesso baseado em biometria facial**. Universidade Federal do Espírito Santo, 2010.

MUELLER John. P.; MASSARON Luca. **Machine Learning:** for dummies. 1 ed. John Wiley & Sons, 2016.

MÜLLER, Andreas C.; GUIDO, Sarah. **Introduction to Machine Learning with Python:** A Guide for Data Scientists. 1 ed. O'Reilly Media, 2017.

NAKASHIRO, Marta M. **Biometria no Brasil e o registro de identidade civil:** novos rumos para identificação. 2011. 126 f. Tese (Doutorado em Sociologia) – Departamento de Pós-Graduação em Sociologia, Universidade do Estado de São Paulo, São Paulo, 2011.

NEOCONTROL. **Casas inteligentes:** chegou a hora de apostar na automação residencial?. Disponível em: <https://www.neocontrol.com.br/news/casas-inteligentes/>. Acesso em: 21 de maio, 2019.

NG, Andrew. **What data scientists should know about deep learning**. Disponível em: <https://www.slideshare.net/ExtractConf>. Acesso em: 17 de novembro, 2019.

NUNES, Jerónimo. **Da Internet para as Pessoas à Internet das Coisas**. 2016. Disponível em: <https://repositorio.uac.pt/handle/10400.3/4205>. Acesso em: 18 de maio, 2019.

NVIDIA. **Embedded Systems for Next-Generation Autonomous Machines**. Disponível em: <https://www.nvidia.com/pt-br/autonomous-machines/embedded-systems>. Acesso em: 8 de dezembro, 2019.

NVIDIA. **OpenCV**. Disponível em: <https://developer.nvidia.com/opencv>. Acesso em: 18 de maio, 2019.

OPENCV. **OpenCV 2.4.13.7 documentation**. Disponível em: < https://docs.opencv.org/2.4/index.html>. Acesso em: 9 de novembro, 2019.

RASPBERRYPI. **Raspberry Pi**. Disponível em: <https://www.raspberrypi.org/documentation/faqs/#introduction>. Acesso em: 18 de maio, 2019.

REMPEL, David; CAMILLERI, Matt J.; LEE, David L. **The Design of Hand Gestures for Human-Computer Interaction:** Lessons from Sign Language Interpreters. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4447613>. Acesso em: 10 de novembro, 2019.

REUTERS. **Amazon's automated grocery store of the future opens Monday**. Disponível em: <https://www.reuters.com/article/us-amazon-com-store/amazons-automated-grocery-store-of-the-future-opens-monday-idUSKBN1FA0RL>. Acesso em: 23 de maio, 2019.

SAHA, Sparcha. **Hand Gesture Recognition Using Background Ellimination and Convolution Neural Network**. Disponível em: <https://github.com/SparshaSaha/Hand-Gesture-Recognition-Using-Background-Elllimination-and-Convolution-Neural-Network>. Acesso em: 15 de agosto, 2019.

SAHA, Sumit. **A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way**. Disponível em: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>. Acesso em: 17 de novembro, 2019.

SHALEV-SCHWARTZ, Shai; BEN-DAVID, Shai. **Understanding Machine Learning:** From Theory to Algorithms. 1 ed. Cambridge University Press, 2014.

SHI, Weisong; LI, Youhuizi; XU, Lanyu; ZHANG, Quan. **Edge Computing:** Vision and Challenges. Disponível em: <https://www.researchgate.net/publication/303890546> . Acesso em: 25 de maio, 2019.

SIDEY-GIBBONS, Jenni. A. M; SIDEY-GIBBONS, Chris. J. **Machine learning in medicine:** a practical introduction. BMC Medical Research Methodology, 2019.

TOLBA, A. S.; EL-BAZ, A. H.; EL-HARBY, A. A. **Face Recognition:** A Literature Review, International Journal of Computer, Electrical, Automation, Control and Information Engineering. vol. 2, no. 7, 2008.

TURNKEY LINUX. **Node.js**: Asynchronous Javascript Framework. Disponível em: <https://www.turnkeylinux.org/nodejs>. Acesso em: 18 de maio, 2019.

WATSON, Stephanie. **How Fingerprinting Works**. Disponível em: <https://science.howstuffworks.com/fingerprinting.htm>. Acesso em: 4 de maio, 2019.

YOUSEG. **Consumo de banda de câmeras IP**. Disponível em: <https://youseg.wixsite.com/youseg/single-post/2017/05/08/Consumo-de-banda-de-cameras-ip>. Acesso em: 16 de novembro, 2019.